XXX Conferencia Latinoamericana en Informática
http://clei2004.spc.org.pe/

Arequipa, Perú
27 Septiembre - 1 Octubre

Actas de la XXX Conferencia Latinoamericana de Informática (CLEI2004)
XII Congreso Iberoamericano de Educación Superior en Computación
XI Concurso de Tesis de Maestría
CLEI-UNESCO

Editores
Mauricio Solar
David Fernández-Baca
Ernesto Cuadros-Vargas

Resúmenes de XXX Conferencia Latinoamericana en Informática CLEI2004

Libro ISBN: 9972-9876-2-0
Depósito Legal: 0401012004-6564

PRÓLOGO

Esta edición corresponde a los trabajos seleccionados para ser presentados en la XXX Conferencia Latinoamericana de Informática (CLEI2004), el XXII Congreso Iberoamericano de Educación Superior en Computación (CIESC2004) y el XI Concurso de Tesis de Maestría CLEI-UNESCO que se realizan en Arequipa (Perú), desde el 27 de Septiembre al 1 de Octubre de 2004.

Los artículos editados en este volumen es el resultado de un trabajo realizado por una gran cantidad de personas. Se recibieron casi 300 trabajos provenientes de 21 países, principalmente de Latinoamérica, pero muchos trabajos de países europeos. Todos los trabajos fueron sometidos a la evaluación de 3 árbitros, a quienes agradecemos su destacada participación en este proceso. El Comité de Programa estuvo conformado por más de 90 miembros de todos los países latinoamericanos, incluyendo miembros del Reino Unido, Australia, Corea, Canadá, EEUU, etc., quienes realizaron sus evaluaciones con el apoyo de más de 170 colaboradores. Un sincero reconocimiento a todos los miembros del Comité de Programa, y a todos sus colaboradores.

Después del proceso de evaluación se seleccionaron 93 trabajos, un 32% de los sometidos, para ser presentados en Arequipa durante CLEI 2004. La mayoría de los trabajos aceptados provienen de Brasil (un 44%), pero también hay trabajos de Alemania, Argentina, Chile, Colombia, Costa Rica, España, Estados Unidos, Holanda, Italia, México, Paraguay, Perú, Portugal, Sudáfrica, Túnez, Uruguay, Venezuela.

Los tópicos de mayor interés por parte de los autores fueron Inteligencia Artificial (Heurísticas, Multiagentes, Scheduling), Redes Neuronales, Ingeniería de Software (Reingeniería, Métricas, Lenguaje y Desarrollo, Lenguajes de Programación, Calidad, Modelamiento), Redes de Computadores (Seguridad, Agentes Móviles), Web, Bases de Datos, Sistemas Operativos, y Sistema Distribuidos.

Para el XXII Congreso Iberoamericano de Educación Superior en Computación se recibieron 43 trabajos provenientes de 9 países latinoamericano, que fueron evaluados por los 16 miembros internacionales del Comité de Programa de este congreso. Finalmente 15 trabajos fueron seleccionados, correspondiendo al 35% de los sometidos, siendo aceptados trabajos de Argentina, Brasil, Costa Rica, Chile y Perú. El concurso de Tesis de Maestría CLEI-UNESCO recibió 27 trabajos de los cuales se han seleccionado los 3 mejores.

Finalmente, quisiéramos agradecer a todo el Comité Organizador de CLEI2004, especialmente a Ernesto Cuadros-Vargas por su destacado trabajo, y a Adenilso da Silva Simão, por todo su apoyo con el sistema WIMPE.

A todos los participantes, les agradecemos la honra de poder recibirlos en esta su casa y les damos la más cordial bienvenida a la histórica Ciudad Blanca de Arequipa deseándoles una semana muy productiva.

Mauricio Solar
Universidad de Santiago de Chile
Pdte. del Comité de Programa

David Fernández-Baca
Iowa State University, Estados Unidos
Co-Pdte del Comité de Programa

Comité de Conducción

Ricardo Baeza-Yates (*U. de Chile, Chile*)
Rodrigo Cardoso (*U. de los Andes, Colombia*)
José Carlos Maldonado (*ICMC-USP, Brasil*)
Mauricio Solar (*U. de Santiago de Chile, Chile*)

Comité de Programa del 30th CLEI

Adenilso da Silva Simao (*ICMC-USP, Brasil*)
Alberto Valderruten (*U. de A. Coruña, España*)
Alejandro Crema (*UCV, Venezuela*)
Alfredo Matteo (*UCV, Venezuela*)
Andreas Polymeris (*U. de Concepción, Chile*)

Caetano Traina Junior (*ICMC-USP, Brasil*)
Carlos Figueira (*USB, Venezuela*)
Carlos Pon (*U. Católica del Norte, Chile*)
Claudia Linhares (*UFCe, Brasil*)
Cristina Boeres (*U. Federal Fluminense, Brasil*)
David Fernández-Baca (Iowa State U., EEUU, co-presidente)
Edgar Chacón (*U. de Los Andes, Venezuela*)

Enrique Vargas (*U. Católica de Paraguay*)

Ernst Leiss (*U. Houston, EEUU*)
Flor Narciso (*U. de los Andes, Venezuela*)

Gabriela Henning (*GIDSTAD-UTN, Santa Fé, Argentina*)
Gonzalo Acuña (*U. de Santiago de Chile*)

Hernán Astudillo (*U. Técnica Federico Santa María, Chile*)
Ignacio Trejos (*Cenfotec, Costa Rica*)
Ingrid Zukerman (*Monash U., Australia*)
Itana Gimenes (*UEM, Brasil*)
Jesús Ravelo (*USB, Venezuela*)
Jorge Aguirre (*SADIO, Argentina*)
José de Jesús Pérez (*U. Autónoma de Bucaramanga, Colombia*)
Juan Francisco Díaz (*U. Del Valle, Colombia*)

Judith Barrios (*U. de Los Andes, Venezuela*)
Luis Rivera Escriba (*UNEF-RJ, Brasil*)
Manuel Bermudes (*U. de Florida, EEUU*)
Marcelo Ladeira (*UNB, Brasil*)

Maria Clicia Castro (*U. Estadual do Rio de*

Alberto Pardo (*U. de la República, Uruguay*)
Aldo Vechietti (*CERIDE, Argentina*)
Alexander Gelbukh (*Chung-Ang. U., Corea*)
Ana Regina Rocha (*CENTROIN, Brasil*)
Benjamin Barán (*U. Nacional de Asunción, Paraguay*)
Camilo Rueda (*P.U.Javeriana de Cali, Colombia*)
Carlos Juiz (*U. de les Illes Balears, España*)
Catalina Lladó (*U. de les Illes Balears, España*)
Cris Pedregal (*U. New México, EEUU*)
Daniel Fridlender (*SADIO, Argentina*)
Domingo Mery (*PUC de Chile*)

Enrique González (*P.U. Javeriana de Bogotá, Colombia*)
Ernesto Cuadros-Vargas (*Sociedad Peruana Computación, Perú*)
Fernanda Kri (*U. de Santiago de Chile*)
Francisco José Quiles (*U. de Castilla La Mancha, España*)
Gentil Lucena (*U. Católica de Brasilia, Brasil*)

Hércules Prado (*U. Católica de Brasilia, EMBRAPA, Brasil*)
Horacio Leone (*GIDSTAD-UTN, Argentina*)

Inés de Castro Dutra (*UFRJ, Brasil*)
Isabel Besembel (*U. de Los Andes, Venezuela*)
Javier Blanco (*SADIO, Argentina*)
Jonás Montilva (*U. de Los Andes, Venezuela*)
José Carlos Maldonado (*USP, Sao Carlos, Brasil*)
Juan Carlos Augusto (*U. de Ulster, Jordanstown, Reino Unido*)
Juan Guillermo Lalinde (*UEAFIT, Medellín, Colombia*)
Karin Becker (*PUC-RS, Brasil*)
Manoel Mendonça (*UNIFACS, Brasil*)
Marcelo Jenkins (*U. de Costa Rica*)
Marco Antonio Alvarez (*U. Católica Dom Bosco, Brasil*)
Maria Cristina Ferreira (*USP, Sao Carlos, Brasil*)

Revisores Adicionales

Max Chacón, Michael Niemann, Milton Quiroga, Miquel Mascaró, Nazareno Aguirre, Nicolas Anquetil, Nicolas Wolovick, Omar Alimenti, Omar Viera, Osvaldo Gómez, Pablo David Villarreal, Patricia Rayón Villela, Pedro Cuenca, Pedro Linares, Pedro Pinacho, Pere Pau Sancho, Ricardo Contreras A., Ricardo Corrêa, Ricardo González, Ricardo Medel, Ricardo Oscar Rodríguez, Rodrigo Cardoso, Rodrigo Ramos, Rodrigo Santos, Rubby Casallas, Roberto Ferrari, Rogério Eduardo Garcia, Rosana Teresinha Vaccare Braga, Rosângela Penteado, Rudinei Goularte, Santiago R. Acuña Castillo, Sergio Nesmachnow, Simone de Lima Martins, Sofía N. Galicia-Haro, Sylvia da Rosa, Tamara Rezk, Tania Tait, Tatiana Sugeta, Tomeu Serra, Valter Vieira de Camargo, Vicente González, Vinod Rebello, Vitor Santos Costa, Wagner Castilho, William Torrealba, Yuval Marom.

Comité de Programa del XII CIESC

Alberto Restrepo (*EAFIT, Colombia*)
Álvaro Tasistro (*U. de la República, Uruguay*)
Daltro José Nunes (*U. Federal do Rio Grande do Sul, Brasil*)
David Fernandez-Baca (Iowa State U., EEUU, co-presidente)
Guillermo Rodríguez (*Inst. Tecnológico de Monterrey, México*)
Héctor Beck (*U. de Tarapacá, Chile*)
Héctor Antillanca (*U. de Santiago de Chile, Chile*)
Jonas Montilva (*U. de Los Andes, Venezuela*)
Juan Álvarez (*U. de Chile, Chile*)
Marta Patino (*U. Politécnica de Madrid, España*)
Marcelo Jenkins (*U. de Costa Rica, Costa Rica*)
Mauricio Solar (U. de Santiago de Chile, Chile Presidente)
Miguel Jonathan (*U. Federal de Río de Janeiro, Brasil*)
Ricardo Jiménez Peris (*U. Politécnica de Madrid, España*)

XI Concurso CLEI-UNESCO de Tesis de Maestría
Comité de Programa

Gerardo Rubino (INRIA.Rennes-Francia)
Hector Cancela (Instituto de Computación de la Universidad de la República-Uruguay)
José Valdeni de Lima (Universidad Federal do Rio Grande do Sul-Brasil)
Patricia Corbo (Secretaria de CLEI, Universidad ORT-Uruguay)
Regina Motz (Instituto de Computación de la Universidad de la República-Uruguay)

Comité Evaluador

Adair Martins Vilas Boas (Argentina)
Alberto Pardo (Uruguay)
Alfredo Olivero (Argentina)
Alfredo Viola (Uruguay)
Álvaro Freitas Moreira (Brasil)
Ana Bove (Uruguay)
Ana Cristina Benso da Silva (Brasil)
Andrea Rodríguez (Chile)
Aurora Sánchez (Chile)
Bartolomeu Coll (Francia)
Horst von Brand (Chile)
Ignacia Ania (México)
Inés Friss de Kereki (Uruguay)
Irene Loiseau (Argentina)
João Batista de Oliveira (Brasil)
John Atkinson (Chile)
Jose Antonio Bogarin Geymayr (Paraguay)
Jose Luis Gomez Cipriano (Perú)
José Valdeni de Lima (Brasil)
Luca Cernuzzi (Paraguay)

Carlos Coello Coello (México)
Carlos Navarrete (México)
Carlos Pon (Chile)
Cecilia Sanz (Argentina)
Claudio Botazzo (Brasil)
Diana Cukierman (Uruguay)
Domingo Hernández (Venezuela)
Domingo Mery (Chile)
Eduardo Grampin (Uruguay)
Ernst Leiss (EEUU)
Fernando Tinetti (Argentina)
Francisco Jose Mônaco (Brasil)
Gastón Mousques (Uruguay)
Gerardo Parra (Argentina)
Guillerme L. Telles (Brasil)
Guillermo Simari (Argentina)
Hector Cancela (Uruguay)
Hernán Astudillo (Chile)
Luciana Nedel (Brasil)
Luis Barbosa (Portugal)
Luis C. Lamb (Brasil)
Luis Petingi (Uruguay)
Marcelo Jenkins (Costa Rica)
Márcia Borba Campos (Brasil)
Marcos Jose Santana (Brasil)
Maria Bragion de Toledo (Brasil)
Nazareno Aguirre (Argentina)
Rafael Gregorio Gamboa Hirales (México)
Rafael Melgarejo (Ecuador)
Ramón Puigjaner (España)
Raúl A. Trejo Ramírez (México)
Regina Helena Carlucci Santana (Brasil)
Regina Motz (Uruguay)
Rodrigo Fernandes de Mello (Brasil)
Tiago Telecken (Brasil)
Vera Lúcia Strube de Lima (Brasil)

Premio Especial Software Libre CLEI-UNESCO

José Carlos Maldonado, ICMS-USP-Brasil
Cláudio Menezes, Consejero Regional CI, UNESCO-Montevideo-Uruguay

Concurso Peruano de Software Libre

Palomino Valverde (Vision Linux, SPC, Perú)
José Alonso Cárdenas Márquez (Perú)
Gerardo Luis Buitrón Lucero (Perú)
Nicolás C.A. Antezana Abarca (SPC, Perú)

Comité Organizador de CLEI2004

Adenilso da Silva Simão (USP-Brasil) - Servicios Electrónicos
Alberto Borda Díaz (SPC-Perú) - Animaciones
Alex Jesus Cuadros-Vargas (ICMC-USP-Brasil, SPC-Perú) - Servicios Electrónicos
Alfredo Paz Valderrama (SPC-Perú) - Comité Financiero, Enlace Inst. Nac.
Andre Luis dos Santos Domingues (USP-Brasil) - Servicios Electrónicos
Cesar Beltran Castanon (IME-USP-Brasil, SPC, Perú)
Denise León (UTP, Perú)
Edgar Rodríguez (Turismo Tropical, Perú) - Agencia de Viajes
Eduardo Rafael Llapa Rodríguez (EESC-USP-Brasil, SPC-Perú)
Eduardo Tejada-Gamero (U. Of Stuttgart-Alemania, SPC-Perú)
Ernesto Cuadros-Vargas (SPC, Perú **Presidente**)
Fernando Martínez Ortiz (UNSA-Perú) - Edición
Fernando Ramírez Lazo (SPC-Perú) -Producción y Prensa

Guillermo Calderón Ruiz (UCSM-Perú) Logística de Locales
Guillermo Cámara Chávez (UFMG-Brasil, SPC, Perú)
Hector Velarde Bedregal (UCSM-Perú) Logística de Locales
Javier Arce Abarca (SPC-Perú) - Animaciones
Jordan Stúart Rosas Zegarra (SPC-Perú) - Animaciones
José Luis Montoya (UCSP-Perú) - Secretario de Organización
Juan Gutierrez C. (SPC-Perú) - Difusión y Propaganda
Julio Guillermo Paredes Cornejo (GUPAC Int-Perú, SPC-Perú)
Luis Antonio Huerta Llamosas (UNSA-Perú) - Seguridad
Luis Chancayauri (UNSA-Perú) - Comité Culturales
Martín Flores (UNSA-Perú) - Logística de Materiales
Mauricio López Belón (SPC-Perú) - Animaciones
Nicolás Antezana (SPC-Perú) Planificación
Nora La Serna (SPC-Perú) - Representante SPC en Lima
Oscar Arce Abarca (SPC-Perú) - Animaciones
Patricia Herrera Cateriano (USP-Brasil, SPC-Perú)
Percy Huertas Niquen (UNSA-Perú) - Logística de Materiales
Percy Pari (SPC-Perú)
Renee Rivera (UTP, Perú)
Víctor Cornejo (UNSA-Perú) - Protocolo
Waldo Cancino Ticona (USP-Brasil, SPC-Perú)
Wilber Ramos Lovón (SPC-Perú) - Actividades Satélites

Indice de sesiones

Tutorial T1

Algoritmos Genéticos

Charla Plenaria CP1

Tutorial T2

Multiagentes

Scheduling

Tutorial T3

Tutorial T9

Heurísticas

Redes Neurales 1

Charla Plenaria CP2

Seguridad

Reingeniería

Agentes Móviles

Métricas

Tutorial T4

Tutorial T3

Redes Neurales 2

CIESC 1

Web I

CIESC 2

Charla Plenaria CP3

Lenguaje y Desarrollo

CIESC 3

Lenguajes de Programación

CIESC 4

Tesis de Maestría CLEI-UNESCO

Web II

Procesos de Software

Charla Plenaria CP4

Calidad

Sistemas Distribuídos I

Modelamiento

Sistemas Distribuídos II

Tutorial T5

Tutorial T6

Bases de Datos/Sistemas Operativos

Aplicaciones

Charla Plenaria CP5

Tutorial T7

Tutorial T8

Tutoriales/Tutorials

Tutorial: E-Business - aligning your business with technology

Ponente/Speaker: Koos Koen

e-mail: koos@knowinter.com

Wits Business School, University of the Witwatersrand - Johannesburg – Sudáfrica

Abstract

The one-day tutorial will cover the E-Business domains of: Customer Relationship Management, E-Communications and E-Commerce. The main objective of this tutorial is to give participants the opportunity to understand leading edge applications in the field of "Technology Enabled Relationship Management"(TERM) that will maximize corporate revenue and minimize operational costs. The course will cover the business issues, enabling technologies and implementation methodologies. The following agenda points will be followed:

- Today's business drivers.
- Introduction to strategy and technology alignment.
- Functional and Technology concepts and models.
- Introduction to new business technologies.
- Case studies.

Keywords:E-Commerce, E-Communications, Strategy, Customer Relationship Management, E-Business, Customer Interaction Centers

Biografía/Biography

Professor Koos Koen Pr.Eng; B.Sc; B.Eng; FCSSA; MECSA Visiting Professor Graduate School of Business Administration University of the Witwatersrand

ACADEMIC EXPERIENCE Professor Koen represents SA on the Communications Technical Working Group of the International Federation for Information Processing (IFIP) for the last 18 years. This is a research and academic focused body registered under UNESCO. IFIP was involved in publishing many standards in Information Processing and Communications. As member of the technical group he was involved in the development of networking and messaging protocol standards that are today used in modern communications networks such as the Internet. He was also involved in the development of Network Management standards and protocols employed by most networks worldwide. As the SA representative he organized a number of very successful research and state-of-the-art conferences in South Africa. He presented some 40 papers, talks and publications on Data Networking, Telecommunications and Call Centers in South Africa, Australia, Zimbabwe, India, Singapore, Malaysia, Tunisia, and Bulgaria.

BUSINESS CONSULTING Professor Koen consulted to a large number of well-known organizations locally and internationally, focusing on Communications Systems, Customer Relationship Management and e-Business.

AWARDS Professor Koen was selected as Data Communications Personality of the year by the CSSA in 1990. He was awarded a Fellowship by the CSSA for the creation of a world class Data Communications awareness in South Africa. This was achieved by representing South Africa at the International Federation for Information Processing, by organizing international events and by implementing world-class networks in South Africa. He received the Outstanding Service Award from IFIP in September 2001 for his contribution to and involvement with IFIP over the last 18 years.

Tutorial: Software Quality Attributes

Ponente/Speaker: Mario R. Barbacci

e-mail: mrb@andrew.cmu.edu

Software Engineering Institute - CMU – Estados Unidos

Abstract

Software quality is the degree to which software possesses a desired combination of attributes (modifiability, security, performance, availability, etc). In this tutorial we describe a few principles for analyzing a software architecture to determine if it exhibits certain quality attributes. We show how analysis techniques indigenous to various quality attribute communities can provide a foundation for performing software architecture evaluation.

Since attributes can interact or conflict - improving one attribute often comes at the price of worsening one or more of the others - it is necessary to trade-off among multiple software quality attributes at the time the software architecture of a system is specified, before the system is developed.

It is important to point out that we do not aim at an absolute measure of "architecture quality"; rather our purpose is to identify scenarios from the point of view of a diverse group of stakeholders (e.g., the architect, developers, users, sponsors) and to identify risks (e.g., inadequate performance, successful denial-of-service attacks) and possible mitigation strategies (e.g., prototyping, modeling, simulation).

In the tutorial I will describe processes to conduct architecture trade-off analyses developed by the Software Engineering Institute (SEI). The objective of the evaluations is to understand a software architecture's fitness with respect to multiple software quality attributes and to identify sensitivity points, trade-offs, and risks. Sensitivity points are architectural decisions that have significant impact on a quality attribute; trade-off are sensitivity points that affect more than one attribute; risks are potential problem in achieving the desire attributes.

Keywords:Software Engineering, Software Quality Attributes

Biografía/Biography

Mario Barbacci recently retired from the Software Engineering Institute (SEI) at Carnegie Mellon University. He was one of the founders of the SEI where he has served in several technical and managerial positions, including Project Leader (Distributed Systems), Program Director (Real-time Distributed Systems, Product Attribute Engineering), and Associate Director (Technology Exploration Department). Prior to joining the SEI he was a member of the faculty in the School of Computer Science at Carnegie Mellon University. His current research interests are in the areas of software architecture and distributed systems. He has written numerous books, articles, and technical reports and has contributed to books and encyclopedias on subjects of technical interest. Barbacci is a member of the Institute of Electrical and Electronic Engineers (IEEE) and the IEEE Computer Society, a member of the Association for Computing Machinery (ACM), and a member of Sigma Xi. He was the founding chairman of the International Federation for Information Processing (IFIP) Working Group 10.2 (Computer Descriptions and Tools) and has served as chair of the Joint IEEE Computer Society/ACM Steering Committee for the Establishment of Software Engineering as a Profession (1993-1995), President-Elect, President, and Past-President of the IEEE Computer Society (1995-1997), IEEE Division V Director (1998-1999), IEEE TAB Strategic Planning and Research Committee (2000-2002). Barbacci is a Fellow of the Institute of Electrical and Electronic Engineers (IEEE) and the recipient of several IEEE Computer Society Outstanding Contribution Certificates, the ACM Recognition of Service Award, and the IFIP Silver Core Award. Barbacci received bachelor's and engineer's degrees in electrical engineering from the Universidad Nacional de Ingenieria, Lima, Peru, and a doctorate in computer science from Carnegie Mellon.

Tutorial: Metodos de clustering y sus aplicaciones

Ponente/Speaker: Pavel Makagonov, Mikhail Alexandrov, Alexander Gelbukh

e-mail: mpp2003@inbox.ru, dyner1950@mail.ru, www.gelbukh.com

Universidad Tecnologica de la Mixteca – México
Instituto Politecnico Nacional – México

Abstract

Parte 1. Algunos metodos matemáticos y técnicas computacionales para clustering. En la plática se presentarán primero varios planteamientos de los problemas de clustering (agrupamiento), con los ejemplos correspondientes. Se mostrarán diferentes procedimientos de transformación y reducción de datos para el uso en los algoritmos de clustering. Se dará un panorama de los métodos frecuentemente usados para clustering, con énfasis a los métodos visuales. Se presentarán las aproximaciones para la verificación de los resultados de clustering.

Parte 2. Aplicaciones de los algoritmos de clustering a la minería de datos. La plática está dedicada a la experiencia de los autores en la minería de datos (descubrimiento de conocimientos). Nuestro enfoque general se basa en clustering de los datos. Cuando se tienen los datos agrupados, el experto humano puede formular hipótesis sobre las causas ocultas de que los objetos o sus atributos se reúnen en varios grupos. Aquí se usan los métodos para clustering en los subespacios de factores, en las dendritas y en los grafos. Otro enfoque que se usa en nuestro trabajo consiste en la presentación de los datos dinámicos en la forma que permite activar la intuición del experto para analizar estos datos. En particular, se presentarán combinaciones diferentes de componentes principales. Se demostrarán unos ejemplos prácticos de la genética, geología, ecología, administración y política.

Parte 3. Aplicaciones de los algoritmos del clustering a la minería de texto. Entre varios problemas de la minería de texto se considerarán la construcción de los diccionarios orientados al dominio, clustering de los documentos muy cortos y la búsqueda de los líderes en los grupos de documentos. Se presentarán algunos enfoques para el análisis de la distribución de las publicaciones científicas por los años de la publicación, donde se usa la técnica de clustering. La ventaja principal de los algoritmos propuestos es su independencia (o poca dependencia) del idioma. Se demostrarán los ejemplos prácticos del procesamiento de los documentos administrativos así como las publicaciones de medicina, matemática y otras disciplinas.

Keywords:Clustering, Analisis visual, Mineria de datos, Mineria de textos

Biografía/Biography

- Pavel Makagonov es el Profesor Titular del Depto de Posgrado de la UTM. Dr. en Ciencias en Matemática Aplicada (Geofísica). Al terminar su Doctorado en 1966 era el Profesor Titular del Depto de Modelacion Matemática del Instituto Estatal de Geología de Moscú. De 1991 a 2002 era el SubMinistro y el SubDirector del Centro Analítico de la Alcaldia de Moscu. A partir de 2002 trabaja en México.

- Mikhail Alexandrov es Profesor Titular del Centro de Investigación en Computación del IPN. Ph.D. en Matemática y Física. Al terminar su Doctorado en 1982 hasta 1997 era Profesor Titular del Departamento de Modelacion Matemática del Instituto Estatal de Geología de Moscú. Como un experto invitado trabajaba también en el Centro Analítico mencionado. A partir de 1997 trabaja en México.

- Alexander Gelbukh es Profesor Titular y Jefe del Laboratorio de Lenguaje Natural del Centro de Investigación en Computación del IPN. Ph.D. en Ciencias de Computación. Al terminar su Doctorado en 1995 hasta 1997 era profesor en la Universidad Estatal Lomonosov de Moscú y como un experto invitado trabajaba en algunas companias rusas y de EE.UU. A partir de 1997 trabaja en México.

Tutorial: Introduction to Optimizing Compilers

Ponente/Speaker: Markus Mock

e-mail: mock@cs.pitt.edu

Dept. of Computer Science - University of Pittsburgh – Estados Unidos

Abstract

In the last couple of years program optimization by compilers has become crucially important to achieving top system performance. There are two main reasons. First, increasingly programs are written in languages that provide higher levels of abstraction to programmers (with obvious software engineering benefits) but require smarter compilers to be efficiently executed (e.g. Java just-in-time compilers). Second, to achieve excellent performance on modern computer architectures and memory hierarchies, a tight cooperation between compiler and computer architecture is required to achieve good performance. Driven by these two demands, compilers have made great improvements in the past two decades, so that frequently optimized programs run several times as fast as unoptimized ones.

In this tutorial we will cover the basic techniques and approaches underlying this tremendous progress and will also learn about the most important and crucial optimizations performed by optimizing compilers today. After a short review of basic compiler structure and techniques (lexical and syntax analysis) the major part of the tutorial will focus on the analyses (data and control flow analysis), program representations, and transformations performed by optimizing compilers. The tutorial should enable the participants to both integrate the advanced compiler material into their curricula and to write their own optimizing compilers. Programmers in general will also benefit by better understanding what an optimizing compiler can and cannot do, and how to cooperate with it to achieve fast programs.

I Preliminaries

- Lexical analysis
- Syntax analysis
- Parsing and grammar
- Syntax-directed translation
- Type checking
- Storage Allocation

II Optimizing Compiler Technology

- Intermediate Representations
- Run-time Support
- Control Flow Analysis
- Introduction Data Flow Analysis
- Data Flow Lattices
- SSA Form
- Dependence & Alias Analysis
- Global Value Numbering
- Conditional Constant Propagation & Redundancy Elimination
- Loop & Procedure Optimization
- Register Allocation
- Code Scheduling

- Interprocedural Analysis

III Advanced Topics

- Program Specialization
- Run-Time Optimization

Keywords:compilers, program optimization, program analysis, code generation

Biografía/Biography

Dr. Mock is assistant professor in the Department of Computer Science at the University of Pittsburgh. He received his M.S. and Ph.D. degrees from the University of Washington, Seattle (in 1997 and 2002, respectively). His research interests are compilers, program analysis, and optimization. In particular, his research focuses on applications of run-time information in compilation, program optimization and software tools. Dr. Mock is a member of ACM, IEEE Computer Society, the German Computer Society (GI), and the Peruvian Computer Society (SPC).

Tutorial: Sistemas de Tiempo Real

Ponente/Speaker: Rodrigo Santos

e-mail: ierms@criba.edu.ar

Dep. Ing. Elétrica y Computadoras, Universidad Nacional del Sur CONICET Avda. Alem 1253, Bahía Blanca – Argentina

Resumen/Resumo

El estudio de los Sistemas de Tiempo Real se convirtió en una disciplina muy activa en los últimos años con la realización de varias conferencias anuales del máximo nivel académico, por ejemplo IEEE Real Time System Symposium, Euromicro Conference on Real Time Systems, IEEE Real Time and Embedded Technology and Applications Symposium, etc. En la sociedad de Computación del IEEE existe un grupo dedicado al estudio de los sistemas de tiempo real y Kluwer realiza la publicación mensual de una revista dedicada exclusivamente a este tipo de sistemas. Los sistemas de tiempo real abarcan una amplia gama de aplicaciones que van desde las muy críticas, como pueden ser los controladores de vuelo en aviones y naves espaciales, a aplicaciones que de fallar no provocarán catástrofes pero que de todos modos están sujetas a restricciones temporales como pueden ser transmisiones de video o audio en una red de comunicaciones. El curso propuesto tiene el objetivo de introducir a los alumnos en el estudio de los distintos tipos de herramientas que se utilizan para el análisis y el diseño de los sistemas de tiempo real con diferentes características como pueden ser los multitareamonoprocesador, multitarea-multiprocesador, manejo de secciones críticas y recursos compartidos. Al concluir el curso, los alumnos tendrán las herramientas necesarias para analizar la factibilidad de cualquier sistema operando en tiempo real.

Programa Analítico

1. Introducción a los Sistemas de Tiempo Real
   - *a*) Definición
   - *b*) Aplicaciones
   - *c*) Campos de investigación y desarrollo
2. Planificación de Sistemas de Tiempo Real
   - *a*) Disciplinas de prioridades
     1) Rueda Ciclica Justa
     2) Periodos Monotónicos Crecientes
     3) Menor Tiempo al Vencimiento.
   - *b*) Condiciones necesarias y suficientes para la factibilidad de los sistemas.
3. Manejo de recursos compartidos
   - *a*) Inversiones de prioridad y bloqueos
   - *b*) Protocolo de prioridades heredadas
     1) Abrazo mortal
   - *c*) Protocolo techo

Bibliografía [1] Liu, C. y J. Layland, "Scheduling algorithms for multiprogramming in hard real time environments", J. ACM, 20, 1, 1973, pág. 46-61.
[2] Leung, J. y J. Whitehead, "On the complexity of fixed -priority scheduling of real-time tasks", Performance Evaluation, 2, 4,1982, pág. 237-250.
[3] Santos, J. et al, "Priorities and protocols in real-time LANs", Computer Communications, 14, 9, 1991, pág. 507-514.
[4] Santos, J. y J. Orozco, "Rate monotonic scheduling in hard real-time systems", Information Processing Letters, 48, 1993, pág. 39-45.

[5] Cayssials, R., J. Orozco, J. Santos, R. Santos, "Rate monotonic scheduling of real-time control systems with the minimum number of priority levels", Euromicro Conference on Real-Time Systems, 1999, York, pág. 54-59.
[6] Lehoczky, J., L. Sha, y Y. Ding, "The rate monotonic scheduling algorithm: exact characterization and average case behaviour", Proc. IEEE Real Time Systemss Symposium, 1989, 166-171.
[7] Katcher, D., S. Sathaye, y J. Strosnider, "Fixed priority scheduling with limited priority levels", IEEE Trans. on Computers, 44, 9, 1996.
[8] Sha, L., R. Rajkumar, J. Lehoczky, "Priority inheritance protocols: an approach to real-time synchronization", IEEE Trans. on Computers, 39, 9, 1990, pág. 1175-1184.
[9] Santos, J., E. Ferro, J. Orozco, R. Cayssials, "A heuristic approach to the multitaskmultiprocessor assignment problem using the empty-slots method and rate monotonic scheduling", Real-Time Systems, 13, 1997, pág. 167-199.
[10] Tindell, K., A. Burns, A. Wellings, "Allocating hard real-time tasks: an NP-hard problem made easy", Real-Time Systems, 4, 1992, pág. 145-165.
[11] Santos, R., J. Urriza, J. Santos y J. Orozco, "Heuristic Use of Singularities for On- Line Scheduling of Real-Time Mandatory/Reward-Based Optional Systems", Proc. 14th Euromicro Conference on Real-Time Systems, Viena, 2002.
[12] Abeni, L., G. Butazzo, "Integrating Multimedia Aplications in Hard Real-Time Systems", Proc. 19th IEEE Real Time System Symposium, Madrid, 1998.
Palabras Clave/Palavras Chave: Sistemas de Tiempo Real/Real-Time Systems, Planificación/Scheduling, Sistemas Empotrados/Embedded Systems

Biografía/Biography

Rodrigo Martin Santos received the Electrical Engineering degree and the Doctorate in Engineering (supervised by Javier Orozco) from the Universidad Nacional del Sur in 1997 and 2001, respectively. He presently holds a postdoctoral scholarship granted by the CONICET, and has been accepted as Assistant Researcher at the CONICET, in both cases supervised by Javier Orozco. His research interests are stochastic, fault tolerant, heterogeneous and reward based systems, all in the real-time field.At the beginning of 2003, he was a visiting scholar at the Scuola Superiore Sant'Anna, University of Pisa, Italy. He assists in the courses in Introduction to Digital Computers and Digital Computers and Interfaces at the under-graduate level and is co-professor of Real Time Open Dynamic Systems at the graduate level. In February 2004, he has delivered an introductory course on Real-Time Systems at the XI Summer School of Information Sciences, University of Rio IV, Cordoba, Argentina.

Rodrigo Martín Santos se graduó de Ingeniero Electrónico y de Doctor en Ingeniería (dirigido por el Dr. Orozco) en la Universidad Nacional del Sur en 1997 y 2001 respectivamente. En la actualidad, posee una beca postdoctoral del CONICET y tiene aprobado su ingreso en la Carrera de Investigador Científico del CONICET en la categoría de Asistente, en ambos casos bajo la dirección del Dr. Orozco. Sus temas de investigación son la planificación de los sistemas de tiempo real estocásticos, tolerantes a las fallas, heterogéneos, y basados en recompensas. Al principio de 2003 realizó una estadia en la Scuola Superiore Sant'Anna de Pisa, Italia. En su actividad docente asiste en los cursos de grado Introducción a las Computadoras Digitales, y Computadoras Digitales e Interfaces. En postgrado es co-profesor del curso Sistemas Dinámicos Abiertos de Tiempo Real. En febrero de 2004 dictó un curso introductorio a los Sistemas de Tiempo Real en la XI Escuela de Verano de Ciencias informáticas, Universidad de Rio IV, Cordoba, Argentina.

Tutorial: About World Information Technology Forum

Ponente/Speaker: Ramon Puigjaner

e-mail: putxi@uib.es

Universitat de les Illes Balears - Department de Matemàtiques i Informàtica 07071 Palma de Mallorca – España

Abstract

The World Information Technology Forum (WITFOR) is an event, a biennial cycle of state-of-the-art conferences on global trends in information and communication technology that are run under the auspices of International Federation for Information Processing (IFIP). The purpose of the WITFOR Conferences is to help implement information development strategies and projects in developing countries. The conference is a partnership between the hosting government and IFIP. An important outcome is an agenda to bridge the digital divide between the developed and developing nations and empowerment of the common man with the help of ICT Technology.

WITFOR will be organized every second year in co-operation with the member societies and local governments in developing countries. The first WITFOR took place in Vilnius Lithuania from 29 August to 2nd September 2003 Vilnius 2003. The Vilnius declaration is available for download in the news page. The second WITFOR will take place in Gaborone, Botswana from 31st August to 2nd September 2005.

The outcome of the second conference will be the Gaborone Protocol addressing the following themes: Building the Infrastructure, Economic Opportunity, Empowerment and Participation, Health, Education, Environment, Agriculture, Social and Ethical Aspects, which will be submitted to the UNESCO General Conference 2005 for adoption.

Keywords:World Information Technology Forum, WITFOR

Biografía/Biography

Ramon Puigjaner obtuvo el título de Ingeniero Industrial por la Universidad Politécnica de Cataluña (Barcelona, España) en 1964, su grado de Maître ès Sciences Aéronautiques de la Ecole Nationale de l'Aéronautique de París (Francia), su Doctorado en Ingeniería Industrial por la Junta General Calificadora para la Obtención del título de Doctor Arquitecto y Doctor Ingeniero en 1972, y su título de Licenciado en Informática por la Universidad Politécnica de Madrid (España) en 1977, por convalidación de título de Técnico de Sistemas obtenido en el Instituto de Informática de Madrid en 1972.

De 1966 a 1987 repartió su tiempo entre la Universidad Politécnica de Cataluña, donde explicaba e investigaba en Control Automático, Arquitectura de Computadores y Evaluación del Comportamiento de Sistemas Informáticos, y diversos puestos en la Industria, principalmente de 1970 a 1987 en UNIVAC (después SPERRY y finalmente UNISYS) donde estuvo a cargo de la medición y el modelado de sistemas informáticos para su sintonización (tunning) y dimensionamiento (sizing). En 1987 se incorporó al Departamento de Ciencias Matemáticas e Informática de la Universidad de las Islas Baleares (Palma, Baleares, España) donde actualmente es Catedrático de Universidad de Arquitectura y Tecnología de Computadores. Desde esta posición ha actuado como asesor del COOB'92 (Comité Organizador de la Olimpiada Barcelona '92), ha dirigido la participación de la UIB en el proyecto Esprit II COMPLEMENT, en el Esprit IV HELIOS, y en el Esprit IV SUCSEDE, además de otras actividades y proyectos en cooperación con la industria así como proyectos de financiación pública. Actúa con frecuencia como revisor y evaluador de proyectos para la Comisión Europea.

Es el representante español en el Technical Committee 6 "Communications"de la IFIP y miembro de los grupos de trabajo WG 6.3 "Performance Evaluation of Computer Networks", WG 6.4 "Local and Metropolitan Communication Systems", WG 10.3 "Distributed Systemsz chairman del WG6.9 "Developing countries"de la IFIP. Ha sido miembro del Editorial Board del Journal on ComputerCommunications.

Fue Decano de la Facultad de Informática de la Universidad Politécnica de Catalunya desde 1979 a 1982, Decano de la Facultad de Informática de la Universidad de las Islas Baleares desde 1988 a 1999, y ha sido el Director de la Escuela Politécnica Superior de la Universidad de las Islas Baleares desde 1999 hasta 2004. Desde 1998 hasta 2004 ha sido Presidente de la Conferencia de Decanos y Directores de Centros Universitarios de Informática Españoles.

Es autor de un libro sobre evaluación de prestaciones y de más de un centenar trabajos entre capítulos de libros, artículos en revistas y presentaciones (evaluadas y publicadas) en congresos.

Tutorial: Inteligencia ambiental y redes sensoriales y de actuadores

Ponente/Speaker: Ramon Puigjaner

e-mail: putxi@uib.es

Universitat de les Illes Balears - Department de Matemàtiques i Informàtica 07071 Palma de Mallorca – España

Abstract

La inteligencia ambiental es un atractivo nuevo tema en el ámbito de la informática que pretende dar soporte a los usuarios en sus acciones, comunicaciones y tratamiento de la información mediante entornos digitales inteligentes. En el término usuario no hay que restringirlo a usuarios humanos sino que pueden ser sistemas empotrados, robots, etc. Estos entornos toman conciencia de la presencia de los usuarios reconociendo su posición, sus acciones y sus necesidades integrando esta información en un contexto útil, que puede reaccionar adecuadamente. Los sistemas basados en entornos inteligentes se caracterizan por cuatro dimensiones: ubicuidad, conocimiento, inteligencia e interacción natural. Los sistemas de inteligencia ambiental usan tecnologías desarrolladas en otras áreas de entre las que destaca la de las redes inalámbricas sensoriales y de actuadores (Wíreless Sensor and Actor Networks, WSAN).

Las redes de sensores y actuadores que se han hecho realidad al converger la tecnología de los sistemas microelectrónico-mecánicos, las comunicaciones inalámbricas y la electrónica digital.

En consecuencia, el esquema del tutorial será el siguiente:

Primera parte: Inteligencia ambiental

- Introducción a la inteligencia ambiental
- Entornos de aplicación
- Factores a considerar

Segunda parte: Redes sensoriales y de actuadores (WSAN)

- Introducción a las WSAN
- Arquitectura de las WSAN
- Características de las WSAN: prestaciones, fiabilidad, etc.
- Componentes de las WSAN: sensores, actuadores.
- Coordinación sensores-actuadores

Tercera parte: Conclusión

Keywords:Inteligencia ambiental (Context aware systems), Software basado en componentes (Component-based software), Software distribuido (Distribute software), Redes sensoriales y de actuadores (Wireless sensor and actor networks), Redes inalámbricas (Wireless networks), Sistemas móviles (Mobile systems)

Biografía/Biography

Ramon Puigjaner obtuvo el título de Ingeniero Industrial por la Universidad Politécnica de Cataluña (Barcelona, España) en 1964, su grado de Maître ès Sciences Aéronautiques de la Ecole Nationale de l'Aéronautique de París (Francia), su Doctorado en Ingeniería Industrial por la Junta General Calificadora para la Obtención del título de Doctor Arquitecto y Doctor Ingeniero en 1972, y su título de Licenciado en Informática por la Universidad Politécnica de Madrid (España) en 1977, por convalidación de título de Técnico de Sistemas obtenido en el Instituto de Informática de Madrid en 1972.

De 1966 a 1987 repartió su tiempo entre la Universidad Politécnica de Cataluña, donde explicaba e investigaba en Control Automático, Arquitectura de Computadores y Evaluación del Comportamiento de Sistemas Informáticos, y diversos puestos en la Industria, principalmente de 1970 a 1987 en UNIVAC (después SPERRY y finalmente UNISYS) donde estuvo a cargo de la medición y el modelado de sistemas

informáticos para su sintonización (tunning) y dimensionamiento (sizing). En 1987 se incorporó al Departamento de Ciencias Matemáticas e Informática de la Universidad de las Islas Baleares (Palma, Baleares, España) donde actualmente es Catedrático de Universidad de Arquitectura y Tecnología de Computadores. Desde esta posición ha actuado como asesor del COOB'92 (Comité Organizador de la Olimpiada Barcelona '92), ha dirigido la participación de la UIB en el proyecto Esprit II COMPLEMENT, en el Esprit IV HELIOS, y en el Esprit IV SUCSEDE, además de otras actividades y proyectos en cooperación con la industria así como proyectos de financiación pública. Actúa con frecuencia como revisor y evaluador de proyectos para la Comisión Europea.

Es el representante español en el Technical Committee 6 "Communications"de la IFIP y miembro de los grupos de trabajo WG 6.3 "Performance Evaluation of Computer Networks", WG 6.4 "Local and Metropolitan Communication Systems", WG 10.3 "Distributed Systemsz chairman del WG6.9 "Developing countries"de la IFIP. Ha sido miembro del Editorial Board del Journal on ComputerCommunications.

Fue Decano de la Facultad de Informática de la Universidad Politécnica de Catalunya desde 1979 a 1982, Decano de la Facultad de Informática de la Universidad de las Islas Baleares desde 1988 a 1999, y ha sido el Director de la Escuela Politécnica Superior de la Universidad de las Islas Baleares desde 1999 hasta 2004. Desde 1998 hasta 2004 ha sido Presidente de la Conferencia de Decanos y Directores de Centros Universitarios de Informática Españoles.

Es autor de un libro sobre evaluación de prestaciones y de más de un centenar trabajos entre capítulos de libros, artículos en revistas y presentaciones (evaluadas y publicadas) en congresos.

Tutorial: Algorithmic Issues in Hidden Markov Models

Ponente/Speaker: David Fernández-Baca

e-mail: fernande@cs.iastate.edu

Department of Computer Science, Iowa State University, Ames, Iowa 50011 – Estados Unidos

Abstract

A hidden Markov model (HMM) is a stochastic system that can, at any given time, be in one of a finite number of states, each of which emits a symbol with a certain probability. Furthermore, transitions between states occur according to certain probabilities. An observer of a HMM can see the sequence it emits, but not the sequence of states that produced the symbols. A basic problem in HMMs is to infer the most likely sequence of states that resulted in a given observed sequence. HMMs are used in applications ranging from speech recognition to gene identification. For example, in speech recognition the observed symbols are a series of phonemes and the problem is to infer the sequence of words that produced it. In this tutorial, we give an overview of HMMs and discuss some of their applications, with special emphasis on their use in bioinformatics. We then discuss some of the algorithmic issues that arise in conjunction with HMMs. In particular, we consider methods for studying the sensitivity of HMMs to the choice of transition probabilities and for estimating the best parameters for a model. We illustrate these approaches through an application in computational biology: estimating the evolutionary distance between two DNA sequences.

Keywords:Hidden Markov models, Statistical models, Bioinformatics, Computational biology, Algorithms, Evolutionary trees, Sensitivity analysis, Optimization

Biografía/Biography

David Fernández-Baca is a Professor of Computer Science at Iowa State University, where he has been a faculty member since 1986. He obtained the undergraduate degree in Computer Engineering (Ingeniería en Computación) in 1980 from the Universidad Nacional Autónoma de México, the MS in Computer Engineering and the PhD in Computer Science from the University of California, Davis in 1983 and 1986, respectively. His research interests are in computational biology (primarily in evolutionary tree construction) and combinatorial optimization (primarily in sensitivity analysis of optimization problems).

Tutorial: El control de calidad en proyectos de Software Libre

Ponente/Speaker: Gunnar Wolf

e-mail: gwolf@gwolf.org

Proyecto Debian – México
Universidad Pedagógica Nacional – México

Abstract

El desarrollo de Software Libre puede parecer, ante una primera aproximación, caótico y desorganizado. Hay, sin embargo, una gran cantidad de proyectos que cuentan con cientos de desarrolladores dispersos en todo el mundo. ¿Cómo es entonces que estos proyectos han logrado productos de calidad comparable o incluso superior a la de los sistemas propietarios?

Como primer paso para responder a esta interrogante resulta obvio el uso de herramientas colaborativas de desarrollo y seguimiento de fallos. Esto, si bien conforma la infraestructura indispensable para asegurar un control de calidad, no es sino el primer paso.

En esta plática analizaremos cómo es implementado el proceso de control de calidad en el proyecto Debian, y mencionaremos algunos aspectos de cómo es esto realizado en otros proyectos.
Keywords:Software Libre, Debian, Control de calidad

Biografía/Biography

Producto de un aprendizaje netamente autodidacta, Gunnar Wolf ha crecido dentro del mundo del cómputo desde muy niño, hace ya 20 años, iniciándose con el uso de Emacs y TeX. Con el paso de los años ha orientado su carrera hacia la seguridad en cómputo y el software libre. Programador para la UPN, miembro externo del Departamento de Seguridad en Cómputo de la DGSCA-UNAM, ex-asesor del Área de Software Libre de la DGSCA-UNAM, primer desarrollador del proyecto Debian en México, fundador del grupo de trabajo que organiza el Congreso Nacional de Software Libre (CONSOL) en México desde el 2002, e invitado a hablar a diferentes congresos en todo su país y en nuestro continente, es un activista del software libre y de la concientización acerca de la seguridad.

Charlas Plenarias/Plenary Talks

Charla Plenaria: Minería de Consultas en la Web

Ponente/Speaker: Ricardo Baeza-Yates

e-mail: rbaeza@dcc.uchile.cl

Centro de Investigación de la Web, Dpto. de Ciencias de la Computación, Universidad de Chile – Chile

Abstract

User queries in search engines and Websites give valuable information on the interests of people. In addition, clicks after queries relate those interests to actual content. Even queries without answers imply important missing synonims or content. In this talk we show several examples on how to use this information to improve the performance of search engines, to recommend better queries, and to improve the information scent of the content of a Website.
Keywords:Minería de la Web, Buscadores Web

Biografía/Biography

Ricardo Baeza-Yates is professor and chair of the CS department of the University of Chile. He is also director of the Center for Web Research, a project funded by the Millennium Scientific Initiative. He obtained a Ph.D. in CS from U. of Waterloo, Canada, 1989. He has been president of the Chilean Computer Science Society (SCCC) from 1992 until 1995, being elected again for 1997-98. During 1993, he received the Organization of American States award for young researchers in exact sciences. In 1994 he received the award to the best engineering research in the last 4 years from the Institute of Engineers of Chile. In 1997 with two Brazilian colleagues obtained the COMPAQ prize to the best Brazilian research article in CS. He was elected to the IEEE CS Board of Governors for the period 2002-04. In 2002 he was appointed to the Chilean Academy of Sciences, being the first person from CS to achieve this position in Chile. His research interests include information retrieval, algorithms, and information visualization. He is co-author of the book Modern Information Retrieval, published in 1999 by Addison-Wesley, as well as co-author of the 2nd edition of the Handbook of Algorithms and Data Structures, Addison-Wesley, 1991; and co-editor of Information Retrieval: Algorithms and Data Structures, Prentice-Hall, 1992.

Charla Plenaria: Carreras de Pre-Grado en Computación: Perfiles Profesionales

Ponente/Speaker: Daltro Nunes

e-mail: daltro@inf.ufrgs.br

Instituto de Informática-Universidade Federal do Rio Grande do Sul – Brasil

Abstract

La computación es una área nueva cuando la comparamos con otras áreas. Por ese motivo, muchos conceptos son ambiguos y temporales. Lo que nosotros conocemos por computador, en Alemania es llamado de calculador (Rechner) e en Francia de ordenador (Ordenateur). En algunos países esta área es conocida como Ciencia de la Computación y en otros como Informática. Esas denominaciones también se llevan a las carreras de esta área. Las denominaciones de Informática y Ciencia de la Computación, en un mismo país, a veces son sinónimos y a veces presentan semánticas distintas.

En cada país la realidad de la implementación de carreras del área de computación es diferente pues la misma depende mucho del modelo de educación adoptado. Entre tanto, el concepto de pre-grado (undergraduate) en computación es adoptado en la mayoria de los países.

La dificultad del entendimiento de lo que es computación ha llevado a muchas implementaciones distintas de carreras. Muchas carreras de pre-grado poseen planes curriculares semejantes pero presentan denominaciones distintas y otros casos presentan el mismo nombre pero con orientaciones completamente diferentes. Algunas carreras presentan la misma denominación pero difieren en el tiempo de duración.

Muchas instituciones usan nombres alternativos o combinaciones de nombres orientados al marketing y con el claro objetivo de ser atractivos para nuevos estudiantes. De esa forma se esta creando una cultura de carreras de "marca". Cada institución ofrece su carrera como una "marca". Es mas o menos lo que sucede con los productos farmacéuticos. Remedios con el mismo principio activo son presentados con nombres diferentes y precios distintos dependiendo del laboratorio que los produce. Sin embargo, es el principio activo que deberia ser usado para determinar la denominación de la carrera.

En esta charla se procurará discutir los problemas de las carreras y perfiles con base en literatura actual y en la experiencia de algunos países dividiédolos en "académicosz "profesionalizantes"estableciendo algunos principios como base.

Keywords:Ciencia de la Computación, Informática, Carreras, Planes Currículares, Perfiles de Profesionales

Biografía/Biography

- Ingeniero Eletricista - Eletrónica por la UFRGS-Brasil
- Master en Ciências em Informática por la PUC/RJ
- Doctor en Informática por la Universidad de Stuttgart-Alemanha
- Coordinador do Post-Grado en Ciencia de la Computación/UFRGS-Brasil
- Coordinador de la carrera de Ciencia de la Computación/UFRGS-Brasil
- Secretario de Educación de la Sociedad Brasileña de Computación-SBC
- Miembro do Comité Asessor del Ministerio de Educación para el área de Computación e Informática
- Miembro del Comité Asessor de Cooperación Internacional de la "Fundação de Amparo a Pesquisa do Rio Grande do Sul"

Charla Plenaria: Calidad y Mejoramiento de Procesos Ágiles de Software

Ponente/Speaker: Marcello Visconti

e-mail: visconti@inf.utfsm.cl

Universidad Técnica Federico Santa María – Chile

Abstract

Desde hace unos pocos años ha habido un interés creciente en las metodologías ágiles (léase "livianas") de software. Caracterizadas alternativamente como antídoto a la burocracia (que correspondería a las metodologías "pesadas.º con énfasis en el proceso) o licencia para hackear (falta de metodologías o sin un proceso identificable) han suscitado gran interés en la industria de software, por su potencialidad para enfrentar aparentemente de mejor manera la aparente incompatibilidad entre requerimientos cambiantes, tiempos de desarrollo escasos, y clientes y usuarios cada vez más exigentes en cuanto a la calidad de los productos de software que demandan y reciben. En esta charla se analizarán algunas claves y desafíos que presentan los métodos ágiles desde la perspectiva de la calidad y el mejoramiento de los procesos de software que sugieren un balance entre agilidad y proceso, entre dinamismo y disciplina como una forma de potenciar la adopción de dichos métodos en la industria de software.
Keywords:Métodos ágiles, Calidad de software, Mejoramiento de procesos de software

Biografía/Biography

Marcello Visconti es Ingeniero Civil Informático de la Universidad Técnica Federico Santa María (UTFSM), Chile, y Doctor en Ciencias de Computación de Oregon State University, USA. Se desempeña como Académico del Departamento de Informática de la UTFSM, desarrollando investigación, docencia y extensión en ingeniería de software, particularmente en temas de calidad de software y mejoramiento de procesos de software. Actualmente es director de la Sociedad Chilena de Ciencias de Computación (SCCC), y representante de Chile ante el Centro Latinoamericano de Estudios en Informática (CLEI). Además, es miembro de la ACM, y de la IEEE Computer Society.

Charla Plenaria: Algunas Técnicas para el Procesamiento de Texto Basadas en Diccionarios

Ponente/Speaker: Alexander Gelbukh

e-mail: www.gelbukh.com

Instituto Politecnico Nacional – México

Abstract

En la plática se presentará un conjunto de los trabajos realizados recientemente por el grupo del Laboratorio de Lenguaje Natural dirigido por el autor.

Problemas. Los principales problemas en el procesamiento y la comprensión de textos en lenguaje natural por computadora se concentran en la resolución de ambigüedades de varios tipos: resolución de ambigüedad de sentidos de palabras (WSD, por las siglas en inglés), resolución de anáfora, y resolución de la ambigüedad sintáctica. El problema consiste en la posibilidad aparente de interpretar una palabra o frase de diferentes maneras, mientras que sólo una de éstas se debe seleccionar como correcta en un texto dado específico. Adicionalmente, como unas aplicaciones específicas de las técnicas desarrolladas, se puede mencionar la detección y corrección de errores en el texto, así como el problema de selección de palabra en la traducción automática.

Algoritmos. Una idea básica que subyace varias técnicas para la resolución de estos problemas es la medida de relación entre las palabras. Se distinguen las relaciones de diferente naturaleza, véase el siguiente párrafo. Dado un problema de ambigüedad y una medida de relación del tipo apropiado, el algoritmo selecciona tal interpretación que maximice la relación de la palabra en cuestión con otras palabras en el texto. Una variante de tal algoritmo es la optimización global de semejanza: encontrar la combinación de las selecciones para cada palabra ambigua en el texto de tal manera que la semejanza total en el texto se maximice. Finalmente, una variante de este algoritmo se aplica a la detección y corrección de errores: si ninguna interpretación de la palabra no es plausible, pero sustituyéndola con una palabra parecida se logra mucho mejor relación con el contexto, se puede sospechar un error y proponer la última palabra como corrección.

Diccionarios. La medida de relación se puede especificar en diferentes diccionarios: de combinaciones de palabras, marcos de subcategorización, escenarios y generalización semántica. En la plática se presentarán las aplicaciones de estos diccionarios a uno o varios problemas arriba formulados usando la simple idea del párrafo anterior.

Herramientas y recursos. Finalmente, se discutirán las herramientas desarrolladas en el Laboratorio que permiten la aplicación, la integración y la compilación (semi)automática de dichos diccionarios, entre éstas: un analizador morfológico y un modo de aprendizaje automático de su diccionario, una técnica para el aprendizaje automático de los marcos de subcategorización y las colocaciones, con o sin generalización semántica, y una tecnología de la compilación de un corpus de textos necesario para esto.

Algunas de las ideas mencionadas se han desarrollados conjuntamente con I. Bolshakov, G. Sidorov, S. Galicia, H. Calvo, R. Morales.

Keywords:Diccionario, Resolución de ambigüedad, Mineria de textos

Biografía/Biography

Alexander Gelbukh es el Profesor Titular y el Jefe del Laboratorio de Lenguaje Natural del Centro de Investigación en Computación del IPN. M.en C. en Matemática de la Universidad Estatal Lomonosov de Moscú y Ph.D. en Ciencias de Computación del Instituto Nacional de Informacion Cientifica y Técnica. Es el miembro de Academia de Ciencias y Sistema Nacional de Investigadores de México, es el autor más que 100 publicaciones en el campo de Linguistica Computacional, Procesamiento de Textos y Métodos Numéricos.

Charla Plenaria: Security and Integrity in Digital Media

Ponente/Speaker: Ernst L. Leiss

e-mail: coscel@cs.uh.edu

Department of Computer Science – University of Houston
Houston Texas – Estados Unidos

Abstract

Digital media are becoming increasingly more common. With this ubiquity come requirements for security (who may gain access the media?) and integrity (is the information unadulterated/original?). Cryptography-based approaches are the traditional means of achieving these objectives. However, they have major problems within the context of the distribution of digital media such a music and video, most notably processing requirements, sensitivity to errors, or increases in the amount of data to be stored or transmitted. One way of overcoming these problems is provided by watermarks.

We start with an introduction to the main objectives, including some observations on data security, to the extent needed for the following. Then we discuss important aspects of digital media, in particular data compression. We given a brief overview of various cryptographic approaches and outline their problems within the context of storing and transmitting digital media. Then we sketch watermarks and show how they can be used to achieve, under certain fairly reasonable assumptions, the stated objectives.
Keywords:Security, Integrity, Watermarks, Intellectual Property

Biografía/Biography

Ernst L. Leiss received graduate degrees in computer science and in mathematics from the University of Waterloo and the Technical University of Vienna. He joined the Department of Computer Science at the University of Houston in 1979. He has lectured in 23 countries and has supervised 13 doctoral dissertations and approximately 100 M.S. theses. Dr. Leiss is author of about 140 peer-reviewed papers; he wrote Principles of Data Security (1982, Plenum), Software Under Siege: Viruses and Worms (1990, Elsevier), Parallel and Vector Computing: A Practical Introduction (McGraw-Hill, 1995), and Language Equations (Springer, 1999). He has contributed articles on data-security and on computer viruses to the Encyclopedia of Physical Science and Technology (1987 and 1990, Academic Press). His research interests range from data security to vector/parallel computing, geophysical data processing, databases, and theory of formal languages.

Artículos de CLEI/CLEI's articles

Uma Metodologia para Auxiliar na Seleção de Atributos Relevantes usados por Algoritmos de Aprendizado no Processo de Classificação de Textos

Claudia A. Martins[1,2] **Maria Carolina Monard**[1] **Edson T. Matsubara**[1]

[1] Universidade de São Paulo
Instituto de Ciências Matemáticas e de Computação
Departamento de Ciências de Computação e Estatística
13560-970, São Carlos, SP, Brazil
e-mail: {cam, mcmonard, edsontm}@icmc.sc.usp.br

[2] Universidade Federal de Mato Grosso
Instituto de Ciências Exatas e da Terra
Departamento de Ciência da Computação
78060-900, Cuiabá, MT, Brazil

Abstract

Existing learning algorithms expect their input to be presented in terms of constrained set of attributes. Thus, learning algorithms cannot be applied directly to the Text Mining task related to text classification which consists in automatically classifying text documents based on their contents.

In order to apply learning algorithms to text classification it is necessary to process the text documents into some form that is acceptable to the chosen algorithm. As every word in a document may be treated as an attribute, the selection of these attributes plays an important role on how well the learning algorithm can generalize.

This work proposes a methodology to select attributes from texts decomposed into words (stems) using the bag-of-words approach, considering the behavior of the learning algorithm used for text classification. The methodology is illustrated using three different algorithms on a set of documents.

Keywords: Text Mining, Preprocessing, Inductive Learning.

Resumo

Os algoritmos de aprendizado existentes utilizam como entrada um conjunto de exemplos descritos como vetores de atributos. Assim, os algoritmos de aprendizado não podem ser aplicados diretamente a tarefas de Mineração de Textos, relacionadas à classificação de textos, que consistem em classificar automaticamente documentos textuais baseado em seu conteúdo.

Na aplicação de algoritmos de aprendizado em classificação de textos é necessário transformar os documentos textuais em um formato aceito pelo algoritmo escolhido. Considerando que toda palavra em um documento pode ser tratado como um atributo, a seleção destes atributos tem uma função importante em quão bem um algoritmo de aprendizado consegue generalizar.

Neste trabalho é proposta uma metodologia para selecionar atributos de textos, decompostos em palavras (*stems*) usando a abordagem *bag-of-words*, considerando o comportamento do algoritmo de aprendizado usado na classificação de textos. A metodologia é ilustrada utilizando três diferentes algoritmos em um conjunto de documentos.

Palavras Chaves: Mineração de Textos, Pré-processamento, Aprendizado Indutivo.

1 Introdução

A tarefa de categorização automática de textos, ou documentos, geralmente emprega técnicas de Aprendizado de Máquina para induzir classificadores de um conjunto de textos rotulados. Entretanto, essa não é uma tarefa trivial devido, principalmente, a forma não estruturada dos textos e a alta dimensão do espaço de possíveis atributos. A transformação dos textos em um formato estruturado, de maneira que possam ser submetidos a algoritmos de Aprendizado de Máquina, tem uma influência fundamental em quão bem um algoritmo de aprendizado consegue generalizar [13]. Essa transformação consiste, basicamente, em identificar e selecionar os atributos a serem utilizados para representar os textos, bem como atribuir valores a esses atributos. Uma outra questão relacionada à tarefa de categorização de textos é a escolha do algoritmo de aprendizado.

Neste trabalho é proposta uma metodologia para execução de experimentos em um processo de Mineração de Textos, com o objetivo de selecionar os atributos que melhor representam os textos considerando a precisão dos classificadores induzidos por algoritmos de aprendizado. Para tanto, é realizado o pré-processamento dos documentos transformando-os em uma tabela atributo-valor e aplicando algumas técnicas para reduzir a dimensionalidade dessa tabela considerando, entre outras coisas, a precisão do classificador induzido pelo algoritmo de aprendizado. O pré-processamento dos documentos é realizado usando uma ferramenta computacional denominada PreTex. A metodologia é ilustrada usando um corpus de documentos e três algoritmos de aprendizado, C4.5rules [12], CN2 [4] e SVMTorch II [5].

Este trabalho está organizado da seguinte forma: na Seção 2 são apresentados resumidamente as fases do processo de Mineração de Textos, apresentando uma visão geral do pré-processamento dos documentos baseada na técnica *bag-of-words* usada neste trabalho; na Seção 3 são descritas as características principais da ferramenta computacional PreTex desenvolvida para realizar o pré-processamento de textos; na Seção 4 é descrita a metodologia proposta, ilustrada na Seção 5 na qual são mostrados os experimentos realizados usando a ferramenta PreTex para selecionar os atributos do corpus de documentos utilizados, bem como os resultados obtidos utilizando diferentes algoritmos de aprendizado; finalmente, na Seção 6 são apresentadas as conclusões.

2 Mineração de Textos

Em Mineração de Textos — MT — algumas fases são essenciais e comuns a qualquer processo, as quais podem ser definidas como: (1) coleta de documentos, (2) pré-processamento de textos; (3) extração do conhecimento e (4) avaliação e interpretação dos resultados.

A primeira fase do processo de MT, coleta de documentos, consiste na recuperação de documentos que são relevantes para o domínio da aplicação do conhecimento a ser extraído. Os documentos coletados devem ser transformados para um formato aceito pelos algoritmos de extração de conhecimento. Esta fase, denominada de pré-processamento de textos, cria uma estrutura que é freqüentemente representada como uma tabela

atributo-valor para a coleção de documentos. Essa fase, apresentada em maiores detalhes nas próximas seções, é computacionalmente cara e um cuidadoso pré-processamento é fundamental para o sucesso de todo o processo de MT.

Os documentos representados em um formato adequado podem ser submetidos a algoritmos de extração de conhecimento com o objetivo de descobrir padrões úteis e desconhecidos nos documentos. E, finalmente, a fase de avaliação verifica se o objetivo foi alcançado ou se algumas das etapas devem ser refeitas.

A seguir, é apresentado resumidamente duas das principais questões relacionadas à fase de pré-processamento: como representar os documentos e como diminuir a dimensionalidade do espaço de atributos.

2.1 Representação de Documentos

Dada uma coleção de documentos $D = \{d_1, d_2, \ldots d_n\}$ e um conjunto de categorias $C = \{c_1, c_2, \ldots c_z\}$ associadas com a coleção de documento D, a tarefa de categorização de textos consiste em induzir um classificador que possa determinar se o documento d_i pertence ou não a categoria c_k, para $i = 1, 2, \ldots n$ e $k = 1, 2, \ldots z$. Os documentos podem ser descritos como vetores na forma $(\mathbf{d}_1, c_1), ..., (\mathbf{d}_n, c_z)$, no qual $\mathbf{d}_i$ é um vetor de alta dimensão representando os termos (palavras) que ocorrem no documento e c_k é a classe associada ao documento.

A identificação dos termos em um documento pode se referir às palavras presentes no texto (*bag of words*), ou em representações mais sofisticadas como frases ou sentenças. Entretanto, resultados experimentais mostraram que representações mais sofisticadas perdem em desempenho com relação a palavras simples [1, 6, 8]. De acordo com [8], a razão mais provável para explicar esses resultados é que, embora termos mais sofisticados tenham qualidade semântica superior, a qualidade estatística é inferior em relação a termos baseados em palavras simples. Assim, pesquisas utilizando representações simples e sofisticadas de documentos continuam ativas.

Cada termo t_j, para $j = 1, 2, \ldots, m$, será um elemento do conjunto de atributos da tabela atributo-valor. A atribuição de valores a cada um dos termos é baseada na freqüência que o termo aparece nos documentos. Dependendo do domínio, a representação binária pode ser adequada para atribuir valores aos atributos. Nesse caso, o valor 1 significa presença do termo j no documento i e o valor 0 ausência do termo. No entanto, a representação binária é muito simples e, geralmente, medidas estatísticas são empregadas levando em consideração a freqüência que um termo aparece no documento, bem como a freqüência que esse termo é encontrado em todos os documentos da coleção de documentos. Por exemplo, *term frequency* (tf) é uma medida que utiliza o número de ocorrências do termo t_j no documento d_i. Porém, quando termos com alta freqüência aparecem em toda (ou na maioria) dos documentos da coleção, esses termos não fornecem informação útil para diferenciar documentos. A medida *inverse document frequency* (idf) favorece termos que aparecem em poucos documentos da coleção. A medida idf, definida como $log\ n/x$, varia inversamente ao número de documentos x que contém o termo t_j em uma coleção de documentos. Assim, pode-se definir a medida $tfidf$ combinando as medidas tf e idf. Essas medidas são apresentadas na Tabela 1.

Tabela 1: Definição das Medidas

Medida	Fórmula	Comentário
tf	$\#(t_j, d_i)$	$\#(t_j, d_i)$ é o número de vezes que o termo t_j ocorre no documento d_i.
$tfidf$	$\#(t_j, d_i).log\frac{n}{x}$	as medidas tf e idf são combinadas, na qual x representa o número de documentos em D em que o termo t_j ocorre pelo menos uma vez.
$tfidfn$	$\frac{tfidf(t_j,d_i)}{\sqrt{\sum_{s=1}^{m}(tfidf(t_s,d_i))^2}}$	um fator de normalização é utilizado na equação $tfidf$ para que documentos de tamanhos diversos sejam tratados com a mesma importância.

Uma outra questão a ser considerada quando se utiliza as medidas tf e idf está relacionada a documentos que possuem um número muito diferente de termos. Em muitas situações, documentos pequenos são representados por poucos termos, enquanto que documentos maiores, geralmente, são representados por muitos termos. Quando uma grande quantidade de termos é usada na representação de documentos, a probabilidade do termo pertencer a um documento é alta e, assim, documentos maiores tem melhores chances de serem relevantes do que documentos menores. Normalmente, todos os documentos relevantes deveriam ser tratados com a mesma importância independente do seu tamanho. Um fator de normalização, nesse caso, deve ser incorporado para igualar o tamanho de vetores dos documentos. A medida $tfidfn$ — Tabela 1 — utiliza o fator de normalização para documentos de tamanhos diversos.

2.2 Redução da Dimensionalidade dos Atributos

Na criação da tabela atributo-valor, cada termo que aparece no documento pode ser um elemento do conjunto de atributos que descreve o documento. Assim, a dimensionalidade do conjunto de atributos é um problema que deve ser tratado. Vários métodos podem ser utilizados a fim de reduzir a quantidade de atributos visando uma melhor representatividade e melhor desempenho do processo de MT. Entre outros, a transformação de cada termo para o radical que o originou, por meio de algoritmos de *stemming*, é um método amplamente utilizado e difundido.

Algoritmos de *stemming*, basicamente, consistem em uma normalização lingüística, na qual as formas variantes de um termo são reduzidas a uma forma comum denominada *stem*. A conseqüência da aplicação de algoritmos de *stemming* consiste na remoção de prefixos ou sufixos de um termo, ou mesmo na transformação de um verbo para sua forma no infinitivo. Portanto, um algoritmo de *stemming* é fortemente dependente do idioma no qual os documentos estão escritos. Um dos algoritmos de *stemming* mais conhecidos é o algoritmo do Porter que remove sufixos de termos em inglês [10]. O algoritmo tem sido amplamente usado, referenciado e adaptado nos últimos 20 anos. Diversas implementações do algoritmo estão disponibilizadas na *Web*, entre elas a página oficial escrita e mantida pelo autor para distribuição do seu algoritmo (http://www.tartarus.org/~martin/PorterStemmer).

A aplicação de algoritmos de *stemming* aos termos dos documentos reduz significativamente a quantidade de possíveis atributos que possam representar os documentos. Porém, na maioria das vezes, essa redução

não é suficiente e outras formas para reduzir a dimensionalidade é necessária. A Lei de Zipf descreve uma maneira de descobrir termos considerados pouco representativos em uma determinada coleção de documentos. A lei, formulada por George Kingsley Zipf professor de lingüística de Harvard (1902-1950), declara que a freqüência de ocorrência de algum evento está relacionada a uma função de ordenação. Zipf mostrou que uma das características das linguagens humanas, populações das cidades e muitos outros fenômenos humanos e naturais, seguem uma distribuição similar, a qual denominou de "*Principle of Least Effort*" [17].

Existem diversas maneiras de enunciar a Lei de Zipf para uma coleção de documentos. A mais simples é procedimental: pegar todos os termos na coleção e contar o número de vezes que cada termo aparece. Se o histograma resultante for ordenado de forma decrescente, ou seja, o termo que ocorre mais freqüentemente aparece primeiro, então, a forma da curva é a "curva de Zipf", para aquela coleção de documentos. Se a curva de Zipf for plotada em uma escala logarítmica, ela aparece como uma reta com inclinação -1. A Lei de Zipf em documentos de linguagem natural pode ser aplicada não apenas aos termos mas, também, a frases e sentenças da linguagem. Na realidade, a lei de Zipf é uma observação empírica que se aplica em diversos domínios, e segue a distribuição $p_1 = c/1, p_2 = c/2, ..., p_n = c/n$, na qual $c = 1/H_n$ e $H_n = \sum_{i=1}^{n} 1/i$ [7]. Ou seja, considerando uma coleção de documentos escritos em linguagem natural, foi observado que o j-ésimo termo mais comum ocorre com freqüência inversamente proporcional a j.

Enquanto Zipf verificou sua lei utilizando jornais escritos em inglês, Luhn [9] usou a lei como uma hipótese nula para especificar dois pontos de corte, os quais denominou de superior e inferior, para excluir termos não relevantes. Os termos que excedem o corte superior são os mais freqüentes e são considerados comuns por aparecer em qualquer tipo de documento, como as preposições, conjunções e artigos. Já os termos abaixo do corte inferior são considerados raros e, portanto, não contribuem significativamente na discriminação dos documentos. Assim, Luhn propôs uma técnica para encontrar termos relevantes, assumindo que os termos mais significativos para discriminar o conteúdo do documento estão em um pico imaginário posicionado no meio dos dois pontos de corte. Porém, uma certa arbitrariedade está envolvida na determinação dos pontos de corte, bem como na curva imaginária, os quais são estabelecidos por tentativa e erro [14]. Como a Lei de Zipf, a técnica não é restrita apenas a termos mas, também, pode ser aplicada a *stem* ou sentenças dos documentos. A seguir, é descrita uma ferramenta computacional, por nós implementada, que utiliza os conceitos apresentados.

3 A Ferramenta PreTex

PreTex é uma ferramenta computacional implementada na linguagem Perl [16] usando o paradigma de orientação a objetos. A ferramenta foi desenvolvida com o objetivo de realizar de forma automática o pré-processamento de uma coleção de documentos escritos em três idiomas distintos: português, espanhol e inglês. A implementação da ferramenta é baseada no algoritmo de *stemming* do Porter para a língua inglesa, o qual foi adaptado para a língua portuguesa e espanhola. A ferramenta também inclui facilidades para

reduzir a dimensionalidade do conjunto de atributos usando a Lei de Zipf e os cortes Luhn.

Resumidamente, dentre as características gerais da ferramenta PreTex, podem ser destacadas algumas, tais como: (i) extrair *stems* de palavras em português, espanhol e inglês; (ii) ignorar palavras que não são consideradas significativas usando uma lista de *stopwords*; (iii) criar arquivos intermediários que contém as freqüências dos *stems* de cada um dos documentos, a freqüência dos *stems* na coleção de documentos e a freqüência das palavras que originam cada um desses *stems*; (iv) utilizar qualquer das quatro medidas definidas na Seção 2.1 para atribuir o valor associado a cada *stem* na coleção de documentos; (v) aplicar a Lei de Zipf e cortes de Luhn; (vi) trabalhar com termos simples ou compostos — 1, 2 e 3-*grams*; (vii) gerar gráficos; (viii) criar a tabela atributo-valor utilizando *stems*.

A lista de *stopwords* padrão de PreTex contém termos gerais tais como artigos, conjunções, preposições, pronomes e alguns advérbios. Essa lista encontra-se armazenada em um arquivo. O usuário pode utilizar somente essa lista de *stopwords* padrão da ferramenta bem como pode criar outros arquivos contendo listas adicionais de *stopwords* específicas do domínio. A ferramenta está preparada para considerar conjuntos de arquivos contendo *stopwords*. Para realizar automaticamente os cortes de Luhn, PreTex tem uma opção para utilizar somente os *stems* que estão no intervalo de freqüência $(\overline{x} - ks; \overline{x} + ks)$ no qual $\overline{x}$ é a média da freqüência dos *stems*, s é o desvio-padrão e k é uma constante definida pelo usuário. Uma outra opção permite ao usuário definir livremente os pontos de corte superior e inferior.

A ferramenta, ilustrada na Figura 1, consiste de dois módulos principais: **Stem.pl** e **Report.pl**. O primeiro módulo é responsável pela transformação de termos nos *stems* correspondentes. A entrada para esse módulo pode ser uma palavra, um documento ou uma coleção de documentos. Na Figura 1 está ilustrado este último caso, no qual a coleção de documentos é identificada pelo nome de um diretório, embaixo do qual encontra-se um conjunto de arquivos tal que cada arquivo contém um dos documentos da coleção. Além disso, o usuário deve especificar o idioma dos documentos, *i.e.* português, inglês ou espanhol e, se for o caso, a lista de *stopwords* por ele definida, a qual será adicionada à lista de *stopwords* padrão da ferramenta. A saída consiste de vários arquivos intermediários descritos resumidamente no item (iii) da descrição das características gerais da ferramenta. Esses arquivos contém informações úteis para o usuário e também são utilizados pelo módulo **Report.pl**.

O módulo **Report.pl** tem como entrada os arquivos intermediários, gerados pelo **Stem.pl**, e um arquivo no qual são especificados os parâmetros de execução. Nesse arquivo é definida qual medida utilizar, os pontos de corte mínimo e máximo de Luhn, bem como a quantidade de *grams* a considerar (termos simples ou compostos). Os valores *default* da ferramenta são: a medida tf, sem corte (todos os *stems* são considerados) e 1, 2 e 3-*grams* (*stems* simples e compostos). A saída do módulo **Report.pl** consiste dos arquivos de dados `.data` e `.names` no formato utilizado pelo DISCOVER[1], além de diversos gráficos que mostram a freqüência dos *stems* na coleção de documentos.

[1] A ferramenta PreTex será integrada futuramente ao ambiente DISCOVER, um projeto de pesquisa em desenvolvimento no Laboratório de Inteligência Computacional, LABIC - http://labic.icmc.usp.br, para planejamento e execução de experimentos relacionados com o uso de sistemas de aprendizado no processo de Mineração de Dados e de Mineração de Textos [3, 11].

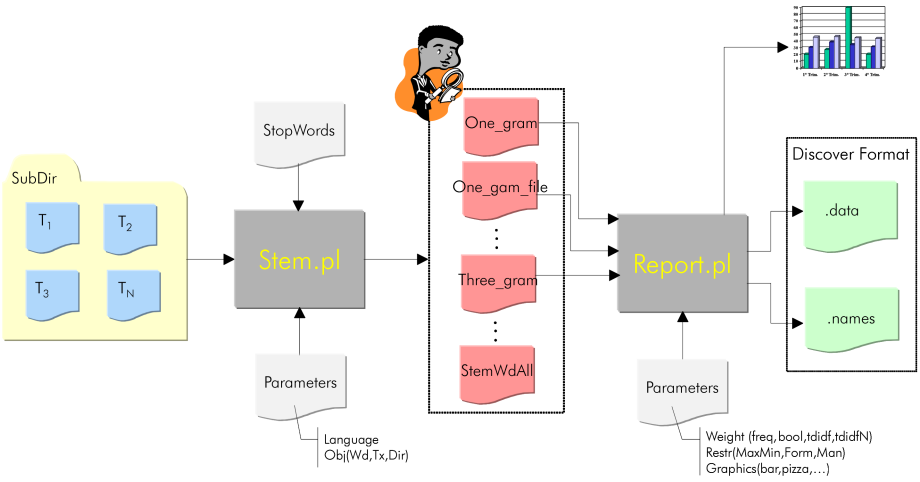


Figura 1: A ferramenta PreTex

4 Metodologia Proposta para Execução de Experimentos

A metodologia proposta é um processo iterativo e interativo e envolve desde a fase de pré-processamento dos textos até a escolha do melhor classificador induzido por algoritmos de aprendizado.

Basicamente, a metodologia consiste em: (i) submeter os documentos ao módulo **Stem.pl** da ferramenta PreTex para encontrar os *stems* dos termos e as freqüências respectivas; (ii) gerar uma ou mais tabelas atributo-valor com o módulo **Report.pl** do PreTex, usando diversas medidas de atribuição de valores aos atributos, eliminando os *stems* com freqüência abaixo de um dado limiar; (iii) encontrar pontos de corte mínimo e máximo usando como referência a quantidade de exemplos da classe minoritária e o desvio padrão da média de freqüência; (iv) submeter as tabelas atributo-valor a algoritmos de aprendizado medindo os erros do classificador induzido utilizando, por exemplo, *10 fold cross-validation*; (v) analisar os erros encontrados pelo classificador.

É proposto um limiar de freqüência, item (ii), para que *stems* com freqüência abaixo desse limiar sejam descartados. Foi definido o valor de 10% da quantidade de exemplos da classe minoritária. Já no item (iii), a proposta é usar num primeiro momento o número de exemplos pertencentes a classe minoritária como referência ao corte mínimo. A idéia de usar esse valor considera a possibilidade de existir um atributo que discrimina perfeitamente uma classe. No caso extremo, esse atributo apareceria somente uma vez em todos os documentos dessa classe, e seu valor mínimo é dado pelo número de documentos na classe minoritária. A partir dos resultados dos experimentos, pode-se definir novos valores para o corte mínimo. Para encontrar o ponto de corte máximo, a nossa proposta é utilizar como referência um ou dois desvios padrão da média de freqüência dos *stems*.

Para ilustrar a metodologia proposta, foram realizados vários experimentos usando uma coleção de documentos escritos em inglês. Essa coleção de documentos, fornecida pelo grupo de pesquisadores do ISISTAN[2], contém 332 documentos classificados em quatro classes: Biomedical, Goats, Music e Sheeps. Cada docu-

[2] http://www.exa.unicen.edu.ar

mento está armazenado em um arquivo texto, extensão `txt`. O tamanho total dessa coleção de documentos é 641,1 KB, cujo tamanho médio dos documentos é 217,47 KB ± 298,33 KB. A Tabela 2 mostra a distribuição desses documentos em cada uma das quatro classes.

Tabela 2: Número de documentos em cada classe

Biomedical	Goats	Music	Sheeps	Total
136 (40,96%)	70 (21,08%)	61 (18,38%)	65 (19,58%)	332 (100%)

Os algoritmos de aprendizado utilizados foram o C4.5rules, CN2 e o SVMTorch II. C4.5rules e CN2 são algoritmos de aprendizado simbólicos que induzem regras de decisão as quais descrevem um contexto específico associado com uma classe. Apesar dos dois algoritmos induzirem regras de decisão, o *bias* indutivo de cada um dos dois algoritmos é muito diferente [2].

SVMs são técnicas de aprendizado baseadas na Teoria de Aprendizado Estatístico proposta por [15]. Essa técnica mapeia os dados de entrada para um espaço abstrato de alta dimensão, onde os exemplos podem ser eficientemente separados por um hiperplano. O SVM incorpora este conceito usando funções denominadas *Kernels*. Essas funções permitem o acesso a espaços complexos de maneira simplificada e computacionalmente eficientes. O hiperplano ótimo nesse espaço é definido como aquele que maximiza a margem de separação entre dados pertencentes a diferentes classes. A principal vantagem dos SVMs é sua precisão e robustez em dados com alta dimensionalidade. Entretanto, diferentemente de algoritmos de aprendizado simbólico, classificadores induzidos utilizando SVMs não são diretamente interpretáveis pelos usuários.

A seguir, a metodologia proposta é ilustrada utilizando a coleção de documentos e os três indutores descritos nesta seção.

5 Resultados Experimentais

A coleção de documentos foi fornecida à ferramenta PreTex, no primeiro passo, especificando como corte mínimo o valor 6, o qual representa 10% da classe minoritária Music com 61 exemplos. Foram encontrados um total de 1284 *stems* com média de freqüência 77,2 ± 85,3. Utilizando tanto os gráficos quanto as tabelas de freqüência de *stems* gerados por PreTex, foi possível observar a distribuição das freqüências dos *stems* na coleção de documentos, Figura 2.

O próximo passo consiste em determinar alguns valores para aplicar os cortes mínimo e máximo, executar PreTex para gerar a tabela atributo-valor correspondente, *i.e.* os arquivos `.data` e `.names` no formato do discover, e observar o erro cometido pelo classificador induzido utilizando essa tabela atributo-valor, de forma a ajustar convenientemente os valores desses cortes.

Na Tabela 3 são mostrados os resultados obtidos em cinco experimentos realizados. Em cada um desses experimentos foram utilizadas duas das quatro medidas implementadas na ferramenta: as medidas tf e $tfidfn$ (Tabela 1) utilizando os algoritmos C4.5rules (identificado por C45r) e CN2. Nessa tabela, Exp identifica o

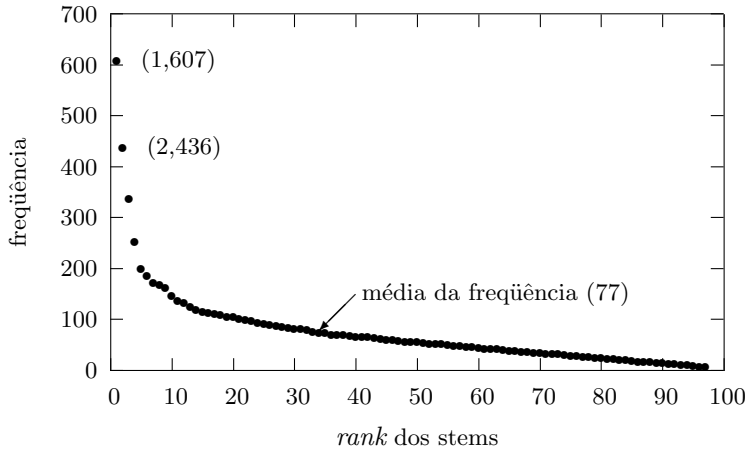


Figura 2: Freqüência de Stems

experimento; Min e Max são, respectivamente, os valores mínimo e máximo utilizados para realizar os cortes de Luhn; # Atr é a quantidade de atributos (*stems*) da tabela atributo-valor construída; Medida identifica a medida utilizada no experimento; A_Erro é o erro aparente dos classificadores induzidos por C4.5rules e CN2, *i.e.* quando toda a coleção de documentos é utilizada para treinar e testar; Erro_10foldCV é o erro e o desvio padrão do classificador induzido calculado usando 10 $fold\ cross\ validation$; # Regras é o número de regras do conjunto de regras que constitui o classificador final e # Atr_Regras é o número de atributos distintos presentes nesse conjunto de regras.

Tabela 3: Resultados experimentais - C4.5rules e CN2

Exp	Min	Max	# Atr	Medida	A_erro		Erro_10foldCV		# Regras		# Atr_R	
					C45r	CN2	C45r	CN2	C45r	CN2	C45r	CN2
				tf	$7,2\%$	$2,7\%$	$6,7\% \pm 1,7\%$	$7,6\% \pm 1,5\%$	5	33	4	54
E_1	6		1284	$tfidfn$	$8,1\%$	$2,7\%$	$13,0\% \pm 1,5\%$	$13,0\% \pm 1,7\%$	8	41	8	69
				tf	$20,5\%$	$3,6\%$	$34,9\% \pm 1,8\%$	$29,8\% \pm 1,9\%$	15	71	18	118
E_2	6	163	1276	$tfidfn$	$20,5\%$	$3,6\%$	$34,1\% \pm 2,1\%$	$27,1\% \pm 1,5\%$	15	71	18	118
				tf	$7,2\%$	$7,5\%$	$12,1\% \pm 1,6\%$	$9,0\% \pm 1,5\%$	9	22	10	29
E_3	61		60	$tfidfn$	$7,8\%$	$10,1\%$	$12,1\% \pm 1,2\%$	$14,1\% \pm 1,7\%$	11	22	17	29
				tf	$8,1\%$	$5,4\%$	$9,3\% \pm 1,6\%$	$9,6\% \pm 1,1\%$	7	24	9	42
E_4	30		172	$tfidfn$	$8,1\%$	$7,5\%$	$16,3\% \pm 1,5\%$	$13,2\% \pm 1,5\%$	10	28	12	48
				tf	$6,3\%$	$4,2\%$	$8,1\% \pm 0,9\%$	$8,2\% \pm 1,5\%$	9	28	10	45
E_5	15		446	$tfidfn$	$8,7\%$	$6,0\%$	$13,9\% \pm 1,2\%$	$11,7\% \pm 1,5\%$	9	32	12	53

No primeiro experimento foram executados os algoritmos com todos os atributos, excluindo apenas os atributos com freqüência menor que 6. Pode ser observado que o erro encontrado por C4.5rules usando a medida tf foi muito bom considerando que o erro da classe majoritária é 59,04%.

O segundo experimento ilustra a busca por um valor de corte Max apropriado. O valor escolhido foi 163. Esse valor está relacionado com a média 77,2 da freqüência dos *stems* mais um desvio padrão. Esse experimento não teve um bom desempenho pois, apesar de ter retirado apenas 8 atributos, pode ser observado que foram retirados atributos relevantes para discriminar as classes, pois o erro incrementou muito. Por exemplo, para a medida tf o erro incrementou aproximadamente 5 vezes (de 6,7% para 34,0%) e quatro vezes (7,6% para 29,8%) para C4.5rules e CN2 respectivamente.

No experimento E_3 foi utilizado como ponto de corte Min o número de exemplos pertencentes a classe minoritária. Pode ser observado que esse valor não é apropriado, visto que o erro aumenta, com relação ao

experimento E_1, para ambos algoritmos. Assim, para os experimentos E_4 e E_5 definiu-se Min com o valor 30 e 15 respectivamente representando, aproximadamente, 50 e 25% do valor anterior. Percebe-se que apesar do erro ter diminuído com relação ao experimento E_3, o erro obtido no experimento E_1 para a medida tf é menor. Levando em consideração somente o erro de classificação, pode-se concluir que os resultados do experimento E_1 são os melhores. Com relação aos modelos induzidos, é possível observar que o mais simples é o induzido por C4.5rules, já que esse classificador consegue resultado semelhante com um conjunto de 5 regras e 4 atributos diferentes, enquanto que CN2 necessita de um conjunto de 33 regras que utilizam 54 atributos diferentes. Na Figura 3 são mostrados os erros dos classificadores induzidos nos experimentos E_1, E_3, E_4 e E_5.

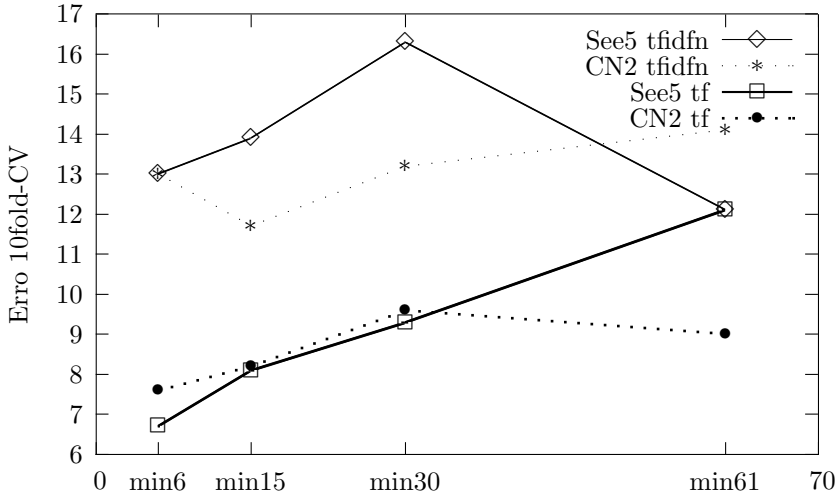


Figura 3: Erros dos Classificadore

Os dados utilizados nos experimentos E_1 e E_5, com os quais foram obtidos os menores erros de classificação, foram submetidos ao algoritmo SVMTorch. Os resultados obtidos são mostrados na Tabela 4, na qual 10fold_errorL representa os erros obtidos usando 10 *fold cross validation* e o *Kernel* Linear; 10fold_errorG representa os erros obtidos usando 10 *fold cross validation* e o *Kernel* Gaussiano com desvio padrão 10.

Tabela 4: Resultados experimentais - SVMTorch

Exp	Min	Max	# Atr	Medida	10fold_errorL	10fold_errorG
E_1	6		1284	tf	$13,5\% \pm 6,5\%$	$17,8\% \pm 5,8\%$
				$tfidfn$	$26,8\% \pm 7,0\%$	$16,3\% \pm 4,1\%$
E_5	15		446	tf	$12,9\% \pm 4,9\%$	$15,0\% \pm 4,4\%$
				$tfidfn$	$23,0\% \pm 9,4\%$	$15,4\% \pm 6,5\%$

Os erros obtidos com SVMTorch foram maiores do que os obtidos com C4.5rules e CN2. Isso mostra mais uma vez a importância de realizar experiências para decidir qual algoritmo de aprendizado é melhor para um determinado conjunto de dados. Como mencionado anteriormente, a técnica de SVMs é considerada muito boa para dados com alta dimensionalidade. Ainda que na maioria dos casos a precisão de SVM é superior à de algoritmos simbólicos, esse resultado não se verifica sempre, como pode ser observado nos experimentos realizados neste trabalho. É interessante observar que na maioria dos experimentos realizados, a medida tf obteve melhores resultados que a medida $tfidfn$. Este é um resultado que também consideramos

particular para este conjunto de dados e indutores utilizados, já que a medida $tfidfn$ não está diretamente correlacionada com a medida tf.

É importante salientar que o uso de um indutor simbólico permite ao usuário/especialista verificar na primeira execução da ferramenta, ou posteriormente, os atributos (*stems*) utilizados pelas regras induzidas. Após, utilizando as informações fornecidas nos arquivos gerados pelo módulo **Stem.pl** da ferramenta PreTex, é possível verificar se esses *stems* correspondem a palavras consideradas relevantes. Caso contrário, essas palavras podem ser colocadas nas listas de *stopwords* do usuário executando novamente a ferramenta.

6 Conclusão

Uma questão primordial em um processo de MT consiste em determinar quais e como serão os atributos que discriminem bem os documentos, visto que a quantidade de possíveis atributos é muito grande. Uma outra questão está relacionada com a escolha do algoritmo de aprendizado a ser utilizado para extrair um bom classificador. Estas são questões que fazem com que a tarefa de MT não seja uma tarefa trivial.

Neste trabalho foi apresentada uma ferramenta computacional cujo objetivo consiste em auxiliar o usuário no pré-processamento de dados textuais. PreTex possui diversos recursos e facilidades para o pré-processamento de documentos utilizando a técnica de *stemming*. Também, foi apresentada uma metodologia para execução de experimentos em MT usando diversos algoritmos de aprendizado e uma mesma coleção de documentos, permitindo assim escolher o melhor classificador para essa coleção de documentos analisando os resultados obtidos.

A fim de ilustrar o uso e da ferramenta PreTex e a metodologia de execução de experimentos foram mostradas algumas experiências realizadas com uma coleção de documentos e três algoritmos de aprendizado.

Agradecimentos

A CAPES e FAPESP pelo auxílio financeiro.

Referências

[1] C. Apté, F. Damerau, and S. M. Weiss. Automated learning of decision rules for text categorization. *Information Systems*, 12(3):233–251, 1994. http://citeseer.nj.nec.com/apte94automated.html.

[2] J. A. Baranauskas and M. C. Monard. An unified overview of six supervised symbolic machine learning inducers. Technical Report 103, ICMC-USP, 2000. ftp://ftp.icmc.sc.usp.br/pub/BIBLIOTECA/rel_tec/Rt_103.ps.zip.

[3] G. E. A. P. A. Batista. Pré-processamento de dados em aprendizado de máquina supervisionado, 2003. Tese de Doutorado, ICMC-USP.

[4] P. Clark and R. Boswell. The cn2 induction algorithm. *Machine Learning*, 3(4):261–283, 1989.

[5] R. Collobert and S. Bengio. SVMTorch: Support vector machines for large-scale regression problems. *Journal of Machine Learning Research*, 1:143–160, 2001.

[6] S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *Proceedings of the 7th International Conference on Information and Knowledge Management (CIKM 98)*, 1998. http://research.microsoft.com/~sdumais/cikm98.pdf.

[7] D. E. Knuth. *The Art of Computer Programming*, volume 3. Addison-Wesley, 1973.

[8] D. D. Lewis. An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of the 15th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 37–50, June 1992.

[9] H. P. Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159–165, 1958.

[10] M. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[11] R. C. Prati. O *framework* de integração do sistema DISCOVER, abril 2003. Dissertação de Mestrado, ICMC-USP.

[12] J. R. Quinlan. *C4.5 Programs for Machine Learning*. Morgan Kaufmann, CA, 1988.

[13] F. Sebastiani. Machine learning in automated text categorisation. *ACM Computing Surveys*, 34(1):1–47, March 2002. http://faure.iei.pi.cnr.it/~fabrizio/Publications/ACMCS02.pdf.

[14] C. J. Van Rijsbergen. *Information Retrieval, 2nd edition*. Dept. of Computer Science, University of Glasgow, 1979. http://citeseer.nj.nec.com/vanrijsbergen79information.html.

[15] V. N. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. theory of probability and its applications. (16):262–280, 1971.

[16] L. Wall, T. Christiansen, and R. L. Schwartz. *Programming in PERL*. O'Reilly,Inc, 1996.

[17] G. Zipf. *Human Behaviour and the Principle of Least Effort*. Addison-Wesley, 1949.

A Genetic Instance-Based Collaborative Approach for Attribute Weighting

Luciana De Nardin
Pontifícia Universidade Católica de Minas Gerais, Dept. de Ciência da Computação,
Poços de Caldas, Brasil, 37701-355
luciana@pucpcaldas.br

Maria do Carmo Nicoletti
Universidade Federal de São Carlos, Dept. de Ciência da Computação,
São Carlos, Brasil, 13565-905
carmo@dc.ufscar.br

Abstract

This paper shows that genetic algorithms can be used as an optimization tool in conjunction with an instance-based learning method, to produce a combination which improves the performance the learning method could achieve on its own. Two instance-based methods are investigated in collaboration with genetic algorithms, namely k-NN and IB2. We conducted a few experiments using a genetic algorithm for finding a 'good' weight vector for either learning algorithms. Classification results on three knowledge domains obtained using k-NN and IB2 modified by a weight vector found by a genetic algorithm, exceeds the performance of the instance-based methods on their own.

Keywords: instance-based methods, lazy learning, genetic instance-based collaboration, weighted NN, weighted IB2.

1. Introduction

Machine Learning is an area of research that provides a vast variety of learning models, algorithms, theoretical results and applications. Lately, a tendency in the development of new learning strategies based on the combination of well-established algorithms can be noticed in the area; the idea is that two learning algorithms can work together to outperform either individually. Among the many possible collaboration methods, one that seems promising is the use of genetic algorithms articulated to instance-based algorithms.

This work focuses on two instance-based algorithms namely, k-NN and IB2 and it is about the use of a genetic algorithm as an optimization tool for finding a weight vector to be used either by k-NN or IB2 aiming at improving their performance when classifying new examples. This paper is organized along the following lines. In Section 2, a general review of instance-based methods and particularly the two algorithms is intended. In Section 3, we describe a few characteristics of genetic algorithms that are relevant to this work and describe how the genetic instance-based collaboration was implemented focusing mainly on the fitness function. In Section 4 we present the main characteristics of the knowledge domains used in the experiments and discuss the results obtained from the collaboration. In the Conclusion, we list the next steps for continuing this work.

2. Instance-Based Learning – Considerations About the Algorithms K-NN and IB2

In contrast to methods that, based on training examples, construct a general description of the concept, instance-based learning methods simply store the training examples. Learning consists of storing the training examples in memory and never changing them. The concept description consists of the training set itself. For classifying a new instance, a distance (possibly weighted) is calculated between the new example and each stored training example and the new example is assigned the class of the nearest neighboring example. A generalization of this procedure takes into consideration the k nearest neighbors and the new example is assigned the class that is most frequent among these k neighbors [8]. The learning phase of these methods consists uniquely of storing; processing happens during classification time.

As commented in [14 – pg. 230] "instance-based methods are sometimes referred to as 'lazy' learning methods because they delay processing until a new instance is classified. A key advantage of this kind of delayed, or lazy, learning is that instead of estimating the target function once for the entire instance space, these methods can estimate it locally and differently for each new instance to be classified."

The nearest neighbor algorithm (NN) [7] is the basis of many lazy learning algorithms. Basically NN techniques assume as the class of an instance x the class of the nearest instance from x. In order to determine the nearest instance, NN techniques adopt a distance metric that measures the proximity of instance x to all stored instances. Figure 1 presents the formal definition of the NN technique found in [9].

It assumes:

- n-dimensional feature space.
- M classes, numbered 1,2,…,M.
- p training instances, each one expressed as a pair (x_i, θ_i), for $1 \leq i \leq p$ where

a) x_i: training instance, expressed by a vector of pairs *attribute-value* $x_i = (x_{i_1}, x_{i_2}, ..., x_{i_{1n}})$

b) $\theta_i \in \{1,2,\ldots,M\}$ expresses the correct class of the instance x_i

Let $T_{NN} = \{(x_1, \theta_1), (x_2, \theta_2), \ldots, (x_p, \theta_p)\}$ be the nearest neighbor training set. Given an unknown instance x, the decision rule is to decide x is in class θ_j if

$$d(x,x_j) \leq d(x,x_i), \text{ for } 1 \leq i \leq p$$

where d is some n-dimensional distance metric.

Figure 1. 1-NN Algorithm

The algorithm described in Figure 1 is more properly called the 1-NN algorithm since it uses only one nearest neighbor. As mentioned earlier, one of the variants of the 1-NN algorithm is the k-NN algorithm, which takes into consideration the k nearest instances $\{i_1, i_2, \ldots, i_k\}$ and decides upon the most frequent class in the set $\{\theta_{i_1},$

θ_{i_2}, …, θ_{i_k}}. Algorithms derived from the nearest neighbor are very popular, mainly due to their simplicity, easy implementation and efficient results.

The k-NN algorithm treats all attributes in a similar way i.e., all the attributes are equally significant. There are situations, however, where the number of significant attributes (significant to the classification process) is small compared to the number of irrelevant attributes. When this happens the large number of irrelevant attributes will dominate the distance between neighbors and they will overcome the truly important attributes. As commented in [14, pg. 231], "…they (instance-based algorithms) typically consider all attributes of the instances when attempting to retrieve similar training examples from memory. If the target concept depends on only a few of the many available attributes, then the instances that are truly most 'similar' may well be a large distance apart."

Associating weights to attributes is a possible way to stress the relevance (or not) of attributes in the expression of the concept. A k-NN algorithm that implements a weight mechanism is generally referred to as Wk-NN.

Instance-based learning methods suffer from several problems and their main disadvantages are related to classification time and memory space, which are proportional to the number of stored examples. The two most relevant decisions to be made concerning these methods are: which training instances should be stored and which distance metric should be adopted in the classification phase, in order to 'measure' the distance of a new example to the stored instances that represent the concept.

Aiming at exploring the limits of instance-based methods, Aha et. al. proposed the IBL (Instance-based Learning) family of algorithms in [1] , which is strongly based on the nearest neighbor algorithm. IBL family groups five algorithms (IB1, IB2, IB3, IB4 and IB5). The first member of the IBL family is IB1 which can be considered the 1-NN algorithm renamed.

Because IB1 stores all training examples and each prediction of a new example involves calculating its distance to each of the stored examples, it becomes very inefficient when the training set becomes large. IB1's storage requirement however, can be reduced without decreasing too much its prediction accuracy by using a storage reduction algorithm from the IBL family, the IB2, which is used in this work. IB2 pseudo code is described in Figure 2.

```
CD ← ∅
for each x ∈ training set do
  begin
   for each y ∈ CD do
     sim[y] ← similarity(x,y)
     y_max ← some y ∈ CD with maximal sim[y]
      if class(x) = class(y_max)
         then classification ← correct
         else begin
                 classification ← incorrect
                 CD ← CD ∪ {x}
              end
  end
```

Figure 2. The IB2 algorithm (CD – Concept Description)

IB2 is identical to IB1 except that it only saves misclassified examples. As commented in [2], "The intuition in IB2's design is that the vast majority of misclassified instances are near-boundary instances that are located in the ε-neighborhood and outside the ε-core of the target concept (for some reasonably small ε)." In spite of IB2 storage reduction capabilities, this algorithm is much more sensitive to the presence of noise in the training set. This sensitivity to noise is a consequence of the fact that during learning, this algorithm only adds to the concept description the training examples that are incorrectly classified. Generally, noisy examples are incorrectly classified and consequently, they tend to be included in the concept description.

This paper is about combining both, k-NN and IB2 with a genetic algorithm aiming at improving the performance of either learning algorithm individually, by means of finding a suitable weight vector, which reflects the real contribution of each attribute that describes the concept. The GA will be used as a procedure that will carry out a search throughout an n-dimensional weight space 'looking for' suitable attribute weight vectors. The goal is to obtain a weight vector such that Wk-NN outperforms k-NN (and correspondently, the weighted version W-IB2 outperforms IB2).

3. Finding a 'Good' Weight Vector – A Contribution Given by a Genetic Algorithm

Although a k-NN which implements a weight strategy tends to have better performance than that which does not, it is very difficult to find a good weight vector. There are a few ways to define a weight vector associated to attributes. The user can define it, based on his/her experience on the knowledge domain. Another possibility is to conduct an exhaustive search throughout the space of all possible weight vectors, trying them all. Depending on the dimension of this space, such a search can be computationally unfeasible. A third option is to use a mathematical tool that could obtain, if not the best, at least a good weight vector which would improve the k-NN (or IB2) performance.

Finding a weight vector can be approached as an optimization problem which can be considered relatively difficult depending on the dimensions of the space to be searched. Several knowledge domains are described by as many as fifty attributes. The problem, in this situation, corresponds to a search for a vector in a 50-dimensional space, where the weight associated to each of the attributes is a real number.

A genetic algorithm (GA) is an adaptive general-purpose search algorithm, which has successfully been applied to many different problems in various areas. The basic principles of GA have been rigorously established by Holland in [12] and can be found in many references (see for instance [3], [4], [10] and [13]).

In GA the term population is used for naming a set of potential solutions to the problem; each individual solution is called a chromosome. Each part of a chromosome (usually representing a variable of the problem) is called a gene. Generally, the initial population is initialized with a pre-defined number of chromosomes which are randomly created; usually the number of individuals per population remains constant during the whole process. Inspired by the biological natural selection process, the GA through selection operator chooses the chromosomes from the current population in order to determine which individual candidates will be part of the 'reproduction' process.

As commented in [6], "Selection attempts to apply pressure upon the population in a manner similar to that of natural selection found in biological systems. Poorer performing individuals are weeded out and better performing, or fitter, individuals have a greater than average chance of promoting the information they contain within the next generation. Crossover allows solutions to exchange information in a way similar to that used by a natural organism undergoing sexual reproduction. Mutation is used to randomly change (flip) the value of single bits within individual strings." The process of selection, crossover and mutation goes on until a convergence criterion has been satisfied. Although there are many different variations of GAs, there is a canonical version, described in Figure 3.

```
procedure GA
begin
  t ← 0
  initialize(p(t))
  evaluate(p(t))
  while not (termination_condition) do
   begin
      t ← t + 1
      select p(t) from p(t-1)
      crossover(p(t))
      mutation(p(t))
      evaluate(p(t))
    end
end
```

Figure 3. Canonical GA

For the problem at hand, the initialization process consists of randomly creating a population of chromosomes, each of them representing a weight vector candidate to be the solution. The dimension of the chromosome is the number of attributes in the domain being considered. In a domain described by N attributes, each chromosome is a vector of N positions, each of them represented by a real number in the interval [0,1]. If the population has been established with size M, then M of such N-dimensional real vectors are randomly created.

The evaluation process uses either (k-NN or IB2) as the fitness function that 'measures' the quality of each chromosome in the population. In order to do that, a 10-fold cross-validation process was implemented; the fitness value of each chromosome is for the average values obtained using the ten learning-testing processes, as shown in Figure 4 and described as a pseudo code in Figure 5.

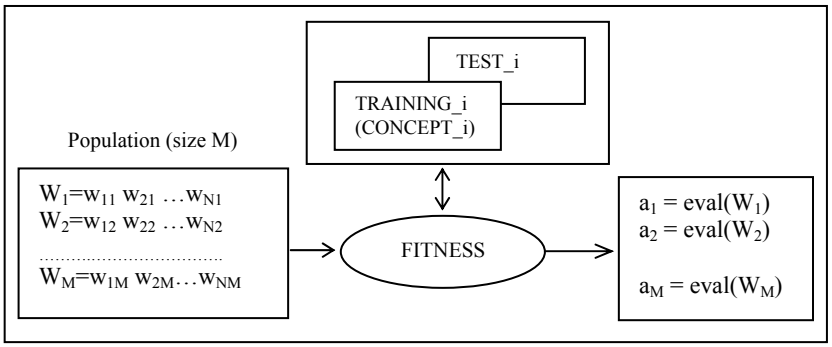


Figure 4. Using the accuracy of a learning algorithm as the fitness function. Each W_i ($1 \le i \le N$) is a weight vector and a_i its corresponding fitness value

The stopping criteria used in the experiments described in the next section was the number of generations; the value of k for implementing the k-NN was 5 (number of neighbors taken into consideration when classifying new examples). The crossover operator implemented is the one-point crossover, which is one of the simplest crossover operators and mutation was implemented as the random operator, which consists in substituting a gene by a random value from its domain.

```
procedure evaluate(P,C_K,T_K);
{P: population (size M) to be evaluated. Chromosome W_I ∈ P (1 ≤ I ≤ M) is a vector
   w_1I w_2I … w_NI , where N is the number of attributes in the domain
 C_K – concept description learnt from training set training_k (1 ≤ k ≤ 10)
 T_K – testing set (corresponding to training_k) to be classified by C_K, taking
       into consideration each W_I ∈ P.
 Each t_p ∈ T_K is a vector described as: t_1p, t_2p, t_3p … t_Np, class(t_p)        }

begin
 for_all W_I ∈ P do
  begin
   number_correct_classif(W_I) ← 0
   for_all t_p = (t_1p t_2p t_3p … t_Np) ∈ T_K do
    begin
     weighting(W_I,t_p,W_It_p)
     classify(C_K,W_It_p,R)
     if R then number_correct_classif(W_I) ← number_correct_classif(W_I) + 1
    end
    aval(W_I) ← number_correct_classif(W_I)/|T_K|
  end
end

weighting(W_I,t_p,W_It_p)
begin
 for q ← 1 to N do W_It_p[q] ← W_I[q]* t_p[q]
end

classify(C_K,W_It_p,R)
begin
 R ← false
 k_NN(C_K,W_It_p,Class)        {classifying with k-NN using a weight}
 if Class = class(W_It_p) then R ← true
end
```

Figure 5. Pseudo code of the evaluation process of a population using k-NN

4. Experiments and Results

The experiments conducted and described in this work are based on data from three knowledge domains, all of them with real-valued attributes. The domains Iris and Wine are well known domains and have been downloaded from the UCI Repository [5] – their descriptions can also be downloaded from the same site. The third domain named Vestibular is a dataset with the results of fixed saccadic tests performed on patients who attended the Service of Otoneurology of the Clinical Hospital which is part of the Medical School of the University of São Paulo, in Ribeirão Preto. The main characteristics of the three domains are shown in Table 1.

The Vestibular System domain data was provided by a medical researcher. Each example is a record for one patient. The data represents the measurement data collected by electrodes which were placed next to the patient's left and right eyes. The movements of both eyes were monitored as they focused on a spotlight, that shone alternatively from one extremity to the other of a horizontal bar, at a constant frequency, during a certain period of time. The electrodes measured the electrical signals which were produced by the saccadic movements. These signals were amplified, filtered and recorded for further analysis. The goal of physicians using these measurements is to be able to detect problems with the Vestibular System of a patient. More information about this domain can be found in [15] and [17].

Table 1. Main characteristics of the domains

Domain	Total Number of Examples	Total Number of Training Examples	Total Number of Testing Examples	Number of Attributes	Number of Classes	Number of Examples per Class
Iris	150	120	30	4	3	50 (setosa) 50 (virginica) 50 (versicolor)
Wine	178	143	35	13	3	59 (Region 1) 71 (Region 2) 48 (Region 3)
Vestibular	199	159	40	6	2	98 (Normal) 101 (Abnormal)

4.1 The Collaboration GA and K-NN in Order to Obtain a WK-NN

In the experiments conducted, we varied the size of the population (50 and 100 individuals) and the number of generations (20, 50, 100 and 500) in order to search for a set of genetic characteristics which would have the best performance. For all the experiments, we used the roulette selection operator, crossover rate of 0.8 and mutation rate of 0.01. The results shown in the next tables are for the best weight vector obtained in the last generation, considering the four different numbers of generations tried.

Iris Domain

Increasing the size of the population did not affect the results; the processing time, however, significantly increased. The significant performance of the population occurred between the 20^{th} and 40^{th} generation; the results of the Wk-NN using GA (obtained using a population size of 50 and generations number of 50) and the k-NN are shown in Table 2.

Table 2. Performance of k-NN versus Wk-NN using GA (Iris domain)

	k-NN	Wk-NN
Correct Classifications (%)	95.94	98.00
Standard deviation value	0.8498364548	0.0005900055

Vestibular Domain

The results in Table 3 showing the performances of both, k-NN and Wk-NN using GA are for size of population 50 and number of generations 50.

Table 3. Performance of k-NN versus Wk-NN using GA (Vestibular domain)

	k-NN	Wk-NN
Correct Classifications (%)	87.00	87.35
Standard deviation value	1.5491933384	0.0001399646

Wine Domain

Table 4 shows the performance values for the k-NN and Wk-NN using GA in the Wine domain, using a population and generation size each of 50. As can be seen in the table, the performance of Wk-NN is considerably inferior to that of the k-NN. In order to explore this domain more, we decided to eliminate the attributes considered less relevant because there was a chance of them interfering negatively in the search process. As suggested in [16], the fifth, sixth, eight and ninth attributes do not contribute much for characterizing the three classes in this domain. Based on this information, we reduced the domain to the nine attributes left and ran the experiments again. As can be seen in Table 5, both algorithms had their performances increased in the reduced domain; the improvement of the Wk-NN, however, was considerably higher compared to its performance on the complete domain.

Table 4. Performance of k-NN versus Wk-NN using GA (Wine domain)

	k-NN	Wk-NN
Correct Classifications (%)	78.88	71.77
Standard deviation value	1.3165610506	0.0032039615

Table 5. Performance of k-NN versus Wk-NN using GA (Reduced Wine domain)

	k-NN	Wk-NN
Correct Classifications (%)	90.00	90.35
Standard deviation value	1.2292725491	0.0012612338

4.2 The Collaboration GA and IB2 for Obtaining a W-IB2

We adopted the same procedure as described in the previous section i.e., in the experiments conducted, we varied the size of the population (50 and 100 individuals) and the number of generations (20, 50, 100 and 500) in order to search for a set of genetic characteristics which would have the best performance. For the experiments, we used the roulette selection operator, crossover rate of 0.8 and mutation rate of 0.01. The results shown in the next tables are for the best weight vector in the last generation, considering the four different numbers of generations tried.

Iris and Vestibular Domains

The results of the W-IB2 using GA (obtained using size of population 100 and 100 generations) and the IB2 are shown in Table 6 for Iris and Vestibular domains. Table 7 shows the best weight vector in the 100th generation for the Vestibular domain.

Table 6. Performance of IB2 versus W-IB2 using GA (Iris & Vestibular domains)

	IB2	W-IB2
	Iris	
Correct Classifications (%)	92.67	95.74
Standard deviation value	0.9944289818	0.0019296422
	Vestibular	
Correct Classifications (%)	77.50	89.98
Standard deviation value	1.7795131356	0.0037664172

Table 7. Best weight vector in the last generation

Number of Attribute	Weight of Attribute
1	0.9886000156
2	0.0610000006
3	0.9006999731
4	0.0653000026
5	0.0586999990
6	0.9714999794

Wine Domain

A similar situation to the one described previously occurred with IB2 in this domain. Table 8, third line, shows the performance values for IB2 and W-IB2 using GA, using the best weight vector in the 100th population, considering the thirteen attributes that describe this domain. Table 8, sixth line, shows the results of both algorithms, using the Reduced Wine domain. As can be seen both algorithms had their performances increased; the improvement of the W-IB2, however, was considerably higher compared to its performance in the complete domain.

Table 8. Performance of IB2 versus W-IB2 using GA (Wine & Reduced Wine domains)

	IB2	W-IB2
	Wine	
Correct Classifications (%)	71.66	66.29
Standard deviation value	2.1317701564	0.0003683525
	Reduced Wine	
Correct Classifications (%)	90.00	92.70
Standard deviation value	1.2292722549	0.0000035762

5. Conclusions

This paper describes how two lazy learning algorithms and a genetic algorithm can collaborate to produce better solutions than either lazy learning algorithm could produce by itself.

Based on the results obtained, we can say that the instance-based learning algorithms have a better performance when they are weighted by a weight vector found with the help of a GA. Although a genetic algorithm does not find optimal weight vectors, it has been shown that it is capable of finding vectors that are good enough to improve the performance of both instance-based algorithms.

In this line of research there are a number of issues that can be addressed in future work, including: to explore more possibilities of the genetic characteristics, such as other selection operators and the creep mutation operator; to focus on class-based attribute weight vectors and to try alternative values for k (in the k-NN). Different values for crossover and mutation rates should also be tried to determine to what extent they interfere in the results. A larger number of knowledge domains should also be considered in future experiments.

Acknowledgements.

To Prof. Dr. José F. Colafemina from the Service of Otoneurology of the Clinical Hospital of the Medical School of University of São Paulo in Ribeirão Preto for proving the Vestibular data and to Leonie C. Pearson whose comments helped us to greatly improve this article.

References

[1] Aha, D. W. Analysis of instance-based learning algorithms. *Proceedings of the 9th National Conference on Artificial Intelligence.* AAAI Press – The MIT Press. Vol. 02 (1991).

[2] Aha, D. W.; Kibler, D. & Albert, M. Instance-based learning algorithms. Machine Learning, 6, (1991), pp. 37-66.

[3] Beasley, D. et. al. An Overview of Genetic Algorithms: Part 1, Fundamentals, University Computing, v. 15, n. 4, (1993), pp. 170-181.

[4] Beasley, D. et. al. An Overview of Genetic Algorithms: Part 2, Fundamentals, University Computing, v. 15, n. 2, (1993), pp. 58-69.

[5] Blake, E. K.C.; Merz, C.J. UCI Repository of machine learning databases, http://www.ics.uci.edu/~mlearn/MLRepository.html, (1998).

[6] Coley, D.A. An Introduction to Genetic Algorithms for Scientists and Engineers, World Scientific, (2001).

[7] Cover, T. and Hart, P. Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory.* Vol. IT 13 (1967), pp. 21-27.

[8] Dasarathy, B.V. (ed.) Nearest neighbour (NN) norms: NN pattern classification techniques. Los Alamitos, CA: IEEE Computer Society Press (1991).

[9] Gates, G. W. "The reduced nearest neighbour rule". *IEEE Transactions on Information Theory*, vol. 18, pp. 431-433, (1972).

[10] Gen, M. and Cheng, R. Genetic Algorithms and Engineering Design. New York, John Wiley (1997).

[11] Goldberg, D. E. Genetic Algorithms in Search, Optimization and Machine Learning. Reading, U.S.A., Addison Wesley Publishing (1989).

[12] Holland, J. H. Adaptation in Natural and Artificial Systems. Ann Arbor, University of Michigan Press (1975).

[13] Michalewicz, Z. Genetic Algorithms + Data Structures = Evolution Programs. Berlin, Springer-Verlag (1992).

[14] Mitchell, T. M. Machine Learning, New York, McGraw-Hill (1997).

[15] Palma Neto, L.G., Figueira, L.B.; Nicoletti, M.C., Using a Family of Perceptron–Based Neural Networks for Detecting Central Vestibular System Problems. In: Proc. of The International Conference on Machine Learning and Applications, M. Wani (ed.), Los Angeles, CA, (2003), pp 193-199.

[16] Ramer, A. and Nicoletti, M. C. The symbolic side of a neuro-fuzzy system. *Studies in Fuzziness and Soft Computing*. P. Sincak and J. Vascak (eds), Physica-Verlag, Heidelberg. Vol. 54 (2000), pp. 447-452.

[17] Volpini, P., Figueira, L. B., Colafemina, J. F., Roque, A. C. A neural network-based system for the diagnosis of central vestibular lesion. In: Valafar, F. (Ed.). Proc. of the Int. Conf. on Mathematics and Engineering Techniques in Medicine and Biological Sciences-METMBS'02, CSREA Press, (2002), pp. 29-33.

Facilitating the Verification of Diffusing Computations and Their Applications

Tanja E.J. Vos
Instituto Tecnológico de Informática
Universidad Politécnica de Valencia, Camino de Vera s/n, 46071 Valencia, Spain
tanja@iti.upv.es

Abstract

We study a class of distributed algorithms, generally known by the name of diffusing computations, that play an important role in all kinds distributed and/or database applications to perform tasks like termination detection, leader election, or propagation of information with feedback. We construct a highly parameterized abstract algorithm and shown that many existing algorithms and their applications can be obtained from this abstract algorithm by instantiating the parameters appropriately and/or refining some of its actions. Subsequently, we show that this use of parameterization and re-usability of notation and proof leads to a reduction of the effort and cost of developing and verifying distributed diffusing computations. More specific, we show that proving the correctness of any application now boils down to verifying an application-specific safety property and reusing the termination and safety proofs of the underlying abstract algorithm.

Keywords: Parameterization, re-use, specifications, formal proof, distributed algorithms, diffusing computations.

1 Introduction

It is our belief that the use of parameterization en re-usability of notation and proof are not sufficiently used when specifying and proving distributed algorithms. This paper uses a family of diffusing computations to show how easily these concepts can be used during the specification of distributed algorithms. We will show how this can lead to better representations of distributed algorithms that: show similarities with (and differences from) related algorithms, increase pedagogical effectiveness, and reduce proof-effort of specific applications of these algorithms. The concepts used in this paper are not new, nor are the algorithms presented. However, specifying a class of distributed algorithms using parameterization and re-usability of notation and showing the many advantages stated above, has, in our opinion, not been sufficiently explored.

2 Diffusing Computations

Diffusing computations, or DCs, were first introduced in [11]. They describe the class of distributed algorithms that first generate a rooted spanning tree (RST) in a connected network of nodes, and then use this tree in order to achieve some goal. Diffusing computations play an important role in computer science since they can be used for: *propagation of information with feedback* [25]; finding the ordering of all node identities in a connected network of nodes [6], and based on this performing *leader election* [29]; computing shortest paths and based on that perform *routing* [2, 15]; *global termination detection* [11, 6, 3, 14, 20, 21, 28, 29]; *deadlock detection* [4, 19] *(re-)synchronization* [12, 24]; *distributed reset* [1]; computing a function of which each node holds part of the input (e.g. summation, maximum finding, distributed approximation of network diameter) [21, 29] and many other *pyramidal functions* [22].

DCs occur under different names in different papers: distributed termination algorithm [14], wave algorithms [29], and total algorithms [28]. Specific instantiations and variations of diffusing computations are TARRY's algorithm, a traversal algorithm for arbitrary networks given by Tarry [27]; ECHO algorithm, introduced by Chang [6]; the classical Depth First Search algorithm (DFS) [7, 29], Segall's PIF [25] algorithms, and the Dijkstra-Scholten Algorithm (DS) [11, 29].

Diffusing computations constitute an interesting case study since they are highly parallel and nondeterministic. Consequently, many have studied them. Less formal work includes [25, 26, 28, 29]. More formal work can be found in [6, 2, 20, 15, 8, 9, 18, 17, 30, 13, 10]. Examining and studying all these works we concluded that specifications and correctness proofs are most of the time not easy to reuse in order to demonstrate the various applications of diffusing computations mentioned above.

3 Terminology and Notation

In this paper we use UNITY because it is simple to explain in a short paper like this. We could, however, have used any action-based formalism. Thus, we do not advocate that UNITY leads to better representations. Each formalism has its advantages and disadvantages, this paper does not go into these details.

UNITY [5] programs Π consist of a predicate (**ini**Π) specifying the initial condition of the program, a set of variable declarations (**v**Π), and a set of actions (**a**Π) separated by the symbol $[\!]$. Actions in a UNITY program (elements of the set Action) can be (multiple) assignments or guarded (if-then-else) actions. Simultaneous execution of assignments is modeled by $\|$, e.g. the action $x := 1 \parallel y := 2$ has the same effect as the multiple assignment $x, y := 1, 2$. A program execution is infinite, in each step an action is selected nondeterministically and executed. Selection is weakly fair, i.e. every action is selected infinitely often.

Program states will be modeled by functions from variables (`Var`) to values (`Val`). State-predicates (elements of Pred) are sets of states modeled by functions from states to bool. State-expressions (elements of Expr) are functions from states to `Val`. When it is clear from the context, operators will be overloaded to denote their state-lifted counterparts.

A state-predicate J is *stable* in Π (denoted ${}_{\Pi}\vdash\circlearrowleft J$), if once J holds during the execution of Π, it will remain to hold. A stable predicate is an *invariant* when it also holds in the initial state of the program.

We will use the original UNITY [5] operators unless (to specify safety properties) and ensures (to specify one-step progress properties). To denote general progress properties, we will use Prasetya's *reach* ($\rightarrowtail$) operator [23]. The most important reason for this choice was the fact that Prasetya has embedded this UNITY variant in the HOL theorem prover [16, 23]. Using this HOL library with its wealth of pre-proved theorems and inference-rules, we saved a lot of time proving the diffusing computations. Details about these mechanical verification activities are outside the scope of this paper and can be found in [32].

Properties involving Prasetya's *reach* operator look like this: $J\ {}_{P}\vdash p \rightarrowtail q$ and, intuitively, mean that if J is stable in program P, then P can make progress from $J \wedge p$ to q. Like leadsto, the *reach* operator is defined as the smallest transitive and disjunctive closure of ensures. However, *reach* requires that the state-predicates

p and q are restricted to be confined by the write variables of program P, which means that $p \rightarrowtail q$ describes progress made through the writable part of a program.

Prasetya [23] also introduced a convergence operator denoted by $J \,{}_{\Pi}\!\vdash p \rightsquigarrow q$. Intuitively, this means that a program Π *converges* from p to q under the stability of J, if, given that ${}_{\Pi}\!\vdash\!\circlearrowleft J$, program Π started in state-predicate p will eventually find itself in a situation where q holds and remains to hold. Laws about $\rightarrowtail$ and $\rightsquigarrow$, needed in this paper, are listed in Appendix 11 and taken from [23].

Function application is represented by a dot, e.g. $s.x$ applies function s to x. Function composition is denoted by $\circ$. In proofs we will, when convenient, replace $\wedge$ by a newline. For example: $p \Leftarrow$ (some theorems and definitions) $q \wedge r$, will be written like:

p
$\Leftarrow$ (some theorems and definitions)
q
r

4 The Communication Network

The communication networks are connected *centralized networks* employing *bi-directional asynchronous communication*. These networks are modeled by a triple ($\mathbb{P}$, neighs, *starter*), where: $\mathbb{P}$ is a finite set of nodes; neighs is a function that given some node $p \in \mathbb{P}$, returns the set of neighbors of p, i.e. the set of nodes that are connected to p by a bi-directional communication link; and *starter* is a node in $\mathbb{P}$ that distinguishes itself from all other nodes (called the *followers*), in that it can spontaneously start the execution of its local algorithm (e.g. because it is triggered by some internal event). The *followers* can only start execution of their local algorithm after they have received a message from some neighbor. Such a network is *connected* if every pair of nodes is connected by a path of communication links.

For this paper it is sufficient to give an abstract model of *asynchronous communication*. We do this by just stating the functionality of the communication primitives. These primitives, listed below, only assign to a specially designated set of Communication Variables, denoted by CV.($\mathbb{P}$, neighs, *starter*):

- mit.$p.q$ (acronym for message in transit) can be used to check for a message in transit from p to q;
- p nr_sent_to q, enables nodes to check how many messages they have already sent to a neighbor q;
- p nr_rec_from q, to check the amount of messages received from q.
- send.$p.q.m$, implements that a node p sends message m to q;
- The receive action contains two subtleties. First, it only has the desired effect when a message is indeed in transit. So the programmer has to ensure that this action is only executed after checking and confirming the availability of a message to receive. Second, when a node p receives a message m from say q, p usually does something with the received value(s) and stores the result(s) somewhere. Since we have no sequential composition of actions in UNITY, receive is parameterized with a function that when applied to p, q and m, returns an assignment that processes the received message and assigns the results to the appropriate variables. So receive.$p.q.a$ implements that p receives a message m from q and processes it using assignment $a.p.q.m$.

Any program that wants to use these communication primitives should incorporate ASYNC_Init.($\mathbb{P}$, neighs, *starter*) in its initial condition in order to initialize the communication variables.

5 Analyzing Diffusing Computations

Most of the diffusing computation algorithms found in literature more or less have the following behavior. A *follower* becomes active when it receives its very first message, and it marks the node from which it received this first message as its *father*. A non-*idle* node proceeds with two activities: *propagation*, i.e. sending a message to all its neighbors except its father; and *collecting* one message from each of its neighbors. When the *starter* has sent and received one message to and from all its neighbors (i.e. has completed *propagating* and *collecting*), it immediately is *done*, whereas a *follower* node p has completed *propagating* and *collecting* it first has to report to its father prior to becoming *done*.

The differences between the algorithms are in the communication protocols, more specifically when non-*idle* nodes are allowed to collect or propagate a message:

prog Π

init ASYNC_Init.($\mathbb{P}$, neighs, *starter*) $\wedge$ $init_\Pi$

var CV.($\mathbb{P}$, neighs, *starter*) $\cup$ $\{p \in \mathbb{P} \mid \mathsf{idle.p}\}$ $\cup$ $\{p \in \mathbb{P} \mid \mathsf{father.p}\}$

assign

$[\!]_{p\in\mathbb{P}}$

$[\!]_{q\in\mathsf{neighs}.p}$ **if** idle.p $\wedge$ mit.$q.p$

then receive.$p.q.\langle$**process**$\rangle$ $\|$ father.$p := q$ $\|$ idle.$p :=$ false (IDLE)

$[\!]$

$[\!]_{q\in\mathsf{neighs}.p}$ **if** $\neg$ idle.p $\wedge$ mit.$q.p$ $\wedge$ $collecting_\Pi.p.q$

then receive.$p.q.\langle$**process**$\rangle$ (COL)

$[\!]$

$[\!]_{q\in\mathsf{neighs}.p}$ **if** $\neg$ idle.p $\wedge$ *can_propagate.p.q* $\wedge$ $propagating_\Pi.p.q$

then send.$p.q.\langle$**mes**$\rangle$ (PROP)

$[\!]$

if *finished_col_and_prop.p* $\wedge$ $\neg$*reported_to_father.p*

then send.$p.$(father.p).$\langle$**mes**$\rangle$ (DONE)

Figure 1: Skeleton of diffusing computation Π.

- The DS algorithm allows a node to freely merge its propagating and collecting actions as long as it has not yet received messages from all its neighbors, and it has not yet sent to all its neighbors that are not its father.
- In the ECHO and PIF algorithms, a non-*idle* node p can only receive a message after p has sent messages to all its neighbors except its father. So, the *propagating* activities must be completed before *collecting* information from non-father-neighbors.
- In the TARRY algorithm, a non-*idle* node p can only propagate to a neighbor if the last event of p was a receive event; otherwise it has to wait until it receives something. So, the *propagating* and *collecting* activities strictly alternate.
- In the DFS algorithm, a non-idle node p in its propagating phase whose last event was receiving a message from some neighbor q: **if** p can propagate a message back to q, i.e. q is not p's father, and p has not yet sent to q, **then** p has to send a message back to this node q **otherwise** it can act like in TARRY, and just pick any non-father-neighbor to which it has not yet sent a message.

5.1 Construct the Abstract Algorithm

Based on the described similarities, we can construct a first skeleton for all local algorithms (including the one for the *starter*). The skeleton is shown in Figure 1. The differences are clearly indicated by sub-scripting the program guards and part of the initial condition of which we have not yet given the exact characterization. Furthermore, the skeleton abstracts from specific applications of the algorithms by leaving the contents of the messages that are being sent, and the ways these are processed upon receipt, unspecified. These two aspects do not only increase the re-usability of the specification, it also gives the reader the opportunity to develop his or her own feeling about the possible uses of the algorithm. This improves pedagogical effectiveness because it can help the reader to better understand the existing applications, and can result in ingenious new ones.

5.1.1 Capturing the Similarities

Besides the similarities revealed by the skeleton in Figure 1, the analysis also taught us that: when a node is *collecting* this implies that it has not yet received messages from all its neighbors**;** when a node is *propagating* this implies that it has not yet sent to all its neighbors that are not its father**;** p *can propagate to* q when p has not yet sent to q, and q is not its father**;** when a node is *finished_col_and_prop*, then it has received from all its neighbors and it has sent to all its non-father-neighbors**;** when a node has not yet reported to its father, it has not yet sent a message to its father **;** when a node p is *done* it has sent and received a message to and from all of its neighbors (i.e. including its father). To capture these similarities we define the following predicates:

$rec\_from\_all\_neighs.p = \forall q \in \mathsf{neighs}.p : p\ \mathsf{nr\_rec\_from}\ q = 1$

$can\_propagate.p.q = \ p\ \mathsf{nr\_sent\_to}\ q = 0\ \wedge\ q \neq \mathsf{father}.p$

$reported\_to\_father.p = \quad (p\ \mathsf{nr\_sent\_to}\ (\mathsf{father}.p) = 1)\ \vee\ (p = starter)$

$sent\_to\_all\_neighs.p = \forall q \in \mathsf{neighs}.p\ : p\ \mathsf{nr\_sent\_to}\ q = 1$

$sent\_to\_all\_non\_fathers.p = \forall q \in \mathsf{neighs}.p : q \neq \mathsf{father}.p\ \Rightarrow\ p\ \mathsf{nr\_sent\_to}\ q = 1$

$finished\_col\_and\_prop.p = rec\_from\_all\_neighs.p\ \wedge\ sent\_to\_all\_non\_fathers.p$

$done.p = rec\_from\_all\_neighs.p \wedge sent\_to\_all\_neighs.p$

5.1.2 Handling the Differences

Differences occur when non-*idle* nodes are allowed to collect or propagate a message, i.e. in the characterization of the *collecting* and *propagating*. In the ECHO and PIF algorithm, the *propagating* activities must be completed before a node can start *collecting* from non-father-neighbors. Consequently:

$propagating_{\text{ECHO}}.p.q = \neg\ sent\_to\_all\_non\_fathers.p$

$collecting_{\text{ECHO}}.p.q = \neg\ rec\_from\_all\_neighs.p\ \wedge\ \neg propagating_{\text{ECHO}}.p.q$

$init_{\text{ECHO}} = \mathsf{father}.starter = starter \wedge \forall p \in \mathbb{P} : p = starter \neq \mathsf{idle}.p$

Instantiating Π with ECHO in Figure 1 specifies the whole ECHO algorithm! The DS algorithm allows a node to freely merge its propagating and collecting actions. Consequently, the *propagating* and *collecting* predicates for this algorithm are:

$propagating_{\text{DS}}.p.q = \neg\ sent\_to\_all\_non\_fathers.p$

$collecting_{\text{DS}}.p.q = \neg\ rec\_from\_all\_neighs.p$

$init_{\text{DS}} = init_{\text{ECHO}}$

Substituting Π for DS in Figure 1 constitutes the DS algorithm.

In the TARRY algorithm, a non-*idle* node p can only propagate to a neighbor if the last event of p was a receive event. We can represent this by introducing a new boolean-typed variable $\mathsf{le\_rec}.p$ (i.e. last event was a receive) for every node p, which we initialize to false for all nodes except the *starter*. In order to give the variable the right value, i.e. whether the last event of p was a receive event, we can add the assignment ($\mathsf{le\_rec}.p := \mathsf{true}$) to the **then** clauses of the actions {IDLE, COL}, and ($\mathsf{le\_rec}.p := \mathsf{false}$) to the **then** clauses of {PROP, DONE}. Finally, we can characterize *collecting* and *propagating* as follows:

$propagating_{\text{TARRY}}.p.q = \neg\ sent\_to\_all\_non\_fathers.p\ \wedge\ (\mathsf{le\_rec}.p)$

$collecting_{\text{TARRY}}.p.q = \neg\ rec\_from\_all\_neighs.p\ \wedge\ \neg(\mathsf{le\_rec}.p)$

$init_{\text{TARRY}} = init_{\text{ECHO}}\ \wedge\ \forall p \in \mathbb{P} : (p = starter) \neq (\neg \mathsf{le\_rec}.p)$

prog Π

init ASYNC_Init.($\mathbb{P}$, neighs, *starter*) $\wedge$ $init_\Pi$

var CV.($\mathbb{P}$, neighs, *starter*) $\cup$ $\{p \in \mathbb{P} \mid \mathsf{idle.p}\}$ $\cup$ $\{p \in \mathbb{P} \mid \mathsf{father.p}\}$

assign

$[\!]_{p\in\mathbb{P}}$
$[\!]_{q\in\mathsf{neighs}.p}$ **if** idle.p $\wedge$ mit.$q.p$

then receive.$p.q.\langle$**process**$\rangle$ $\|$ father.$p := q$ $\|$ idle.$p :=$ false $\boxed{\| \text{IDLE\_a}_\Pi.p.q}$ (IDLE)

$[\!]$

$[\!]_{q\in\mathsf{neighs}.p}$ **if** $\neg$ idle.p $\wedge$ mit.$q.p$ $\wedge$ $collecting_\Pi.p.q$

then receive.$p.q.\langle$**process**$\rangle$ $\boxed{\| \text{COL\_a}_\Pi.p.q}$ (COL)

$[\!]$

$[\!]_{q\in\mathsf{neighs}.p}$ **if** $\neg$ idle.p $\wedge$ $can\_propagate.p.q$ $\wedge$ $propagating_\Pi.p.q$

then send.$p.q.\langle$**mes**$\rangle$ $\boxed{\| \text{PROP\_a}_\Pi.p.q}$ (PROP)

$[\!]$

if $finished\_col\_and\_prop.p \wedge \neg reported\_to\_father.p$

then send.p.(father.p).$\langle$**mes**$\rangle$ $\boxed{\| \text{DONE\_a}_\Pi.p}$ (DONE)

Figure 2: Augmented skeleton to deal with different communication strategies.

In order to adjust the skeleton from Figure 1 we synchronously superpose 4 functions upon it: IDLE_a$_\Pi$, COL_a$_\Pi$, PROP_a$_\Pi$ $\in$ $\mathbb{P} \to \mathbb{P} \to$ Action, and DONE_a$_\Pi$ $\in$ $\mathbb{P} \to$ Action, that specify additional functionality for each action in the skeleton. The result is in Figure 2. We can now complete the characterization of TARRY by defining for all $p, q \in \mathbb{P}$ that IDLE_a$_{\text{TARRY}}.p.q$ and COL_a$_{\text{TARRY}}.p.q$ equal le_rec.p := true, and that PROP_a$_{\text{TARRY}}.p.q$ and DONE_a$_{\text{TARRY}}.p$ equal le_rec.p := false. Note that IDLE_a.$p.q$, COL_a.$p.q$, PROP_a.$p.q$ and DONE_a.p for ECHO and DS are all skip.

In order to be able to formalize the DFS algorithm, we need to be able to remember the identity of the sender of the last incoming message. In order to make this possible, we introduce a new variable lp_rec.p (last process of which p has received a message) for every node p. Then we define for all $p \in \mathbb{P}$ that IDLE_a$_{\text{DFS}}.p.q$ and COL_a$_{\text{DFS}}.p.q$ equal le_rec.p := true $\|$ lp_rec.$p := q$, and that PROP_a$_{\text{DFS}}.p.q$ and DONE_a$_{\text{DFS}}.p$ are le_rec.p := false. Thus, we define:

$propagating_{\text{DFS}}.p.q = propagating_{\text{TARRY}}.p.q \wedge (q = \mathsf{lp\_rec}.p \ \vee \ \neg can\_propagate.p.(\mathsf{lp\_rec}.p))$

$collecting_{\text{DFS}}.p.q = collecting_{\text{TARRY}}.p.q$

$init_{\text{DFS}} = init_{\text{TARRY}}$

6 Applications of Diffusing Computations

Precise discussion of the applications of diffusing computations requires us to specify the contents of the messages that are being sent and the ways these are processed upon receipt. Consequently, we parameterize our skeleton such that specific applications can be defined as *instantiations* of the underlying algorithm, and abstraction from applications is done by *universal quantification* over its arguments. In order to make the algorithms suitable for the characterization of specific applications, they will have to be parameterized with additional parameters:

- a predicate $initA \in$ Expr, which can be used to specify an additional application specific initial condition.

$\Pi.initA.$IDLE_r.COL_r.PROP_mes.DONE_mes.IDLE_a.COL_a.PROP_a.DONE_a.$\mathcal{A}.\mathcal{V}$ =

prog Π

init ASYNC_Init.($\mathbb{P}$, neighs, *starter*) $\wedge$ $init_\Pi$ $\wedge$ $\boxed{initA}$

var CV.($\mathbb{P}$, neighs, *starter*) $\cup$ $\{p \in \mathbb{P} \mid \mathsf{idle.p}\}$ $\cup$ $\{p \in \mathbb{P} \mid \mathsf{father.p}\}$ $\cup$ $\boxed{\mathcal{V}}$

assign

$[\!]_{p\in\mathbb{P}}$

$[\!]_{q\in\mathsf{neighs}.p}$ **if** idle.p $\wedge$ mit.$q.p$

(IDLE)

then receive.$p.q.\boxed{(\text{IDLE\_r}.p.q)}$ $\|$ father.$p := q$ $\|$ idle.$p :=$ false $\|$ IDLE_a$_\Pi.p.q$ $\|$ $\boxed{\text{IDLE\_a}.p.q}$

$[\!]$

$[\!]_{q\in\mathsf{neighs}.p}$ **if** $\neg$ idle.p $\wedge$ mit.$q.p$ $\wedge$ $collecting_\Pi.p$

(COL)

then receive.$p.q.\boxed{(\text{COL\_r}.p.q)}$ $\|$ COL_a$_\Pi.p.q$ $\|$ $\boxed{\text{COL\_a}.p.q}$

$[\!]$

$[\!]_{q\in\mathsf{neighs}.p}$ **if** $\neg$ idle.p $\wedge$ *can_propagate.p.q* $\wedge$ $propagating_\Pi.p$

(PROP)

then send.$p.q.\boxed{(\text{PROP\_mes}.p.q)}$ $\|$ PROP_a$_\Pi.p.q$ $\|$ $\boxed{\text{PROP\_a}.p.q}$

$[\!]$

if *finished_col_and_prop.p* $\wedge$ $\neg$*reported_to_father.p*

(DONE)

then send.$p.$(father.p).$\boxed{(\text{DONE\_mes}.p)}$ $\|$ DONE_a$_\Pi.p$ $\|$ $\boxed{\text{DONE\_a}.p}$

$[\!]$

$\boxed{\mathcal{A}}$

Figure 3: Parameterized skeleton to deal with various applications.

- a function IDLE_r $\in \mathbb{P} \to \mathbb{P} \to$ Expr $\to$ Action, that specifies how an idle node should process messages upon receipt.
- a function COL_r $\in \mathbb{P} \to \mathbb{P} \to$ Expr $\to$ Action, that specifies how a non-idle process should process messages upon receipt.
- a function PROP_mes$\in \mathbb{P} \to \mathbb{P} \to$ Expr, that, given a node p, specifies what message p should send to a neighbor in its propagating phase.
- a function DONE_mes$\in \mathbb{P} \to$ Expr, that, given a node p, specifies which message p finally has to send to its father.
- again 4 functions like: IDLE_a, COL_a, PROP_a$_\Pi$ $\in \mathbb{P} \to \mathbb{P} \to$ Action, and DONE_a $\in \mathbb{P} \to$ Action, that specify additional functionality by synchronous superposition for each action in the skeleton but are independent of the underlying algorithm.
- a set of actions $\mathcal{A}$ that, by ways of asynchronous superposition, can be used to add additional functionality or behavior.
- a set of new variables $\mathcal{V}$ that are assigned by the superposed (synchronous and asynchronous) actions.

Figure 3 shows the result. After discussing termination of diffusing computations in §7, we will show how easy it now becomes to create specifications and prove the correctness of specific applications of diffusing computations.

7 Termination of Diffusing Computations

Termination of diffusing computations means that when the algorithm is started in the initial state, eventually each node will reach the situation in which it neither sends nor receives any more messages, and all communication channels will be empty. Termination is independent of the contents of the messages that are being sent and how these are processed upon receipt. Suppose we have arbitrary *initA*, IDLE_r, COL_r, PROP_mes, DONE_mes, IDLE_a, COL_a, PROP_aand DONE_a. Let us abbreviate, for some $\Pi \in \{$DS, ECHO, PIF, TARRY, DFS$\}$:

$T_\Pi = \Pi.initA.\text{IDLE\_r}.\text{COL\_r}.\text{PROP\_mes}.\text{DONE\_mes}.\text{IDLE\_a}.\text{COL\_a}.\text{PROP\_a}.\text{DONE\_a}.\mathcal{A}$

Termination for this family T_Π of algorithms, for some invariant J_Π of Π, is specified as:

$$J_\Pi \ {}_{T_\Pi}\vdash \ \mathbf{ini}T_\Pi \rightsquigarrow (\forall p : p \in \mathbb{P} : done.p) \tag{1}$$

Evidently, when this specification has been proved for some invariant J_Π of Π, it can be inferred for all possible combinations of *initA*, IDLE_r, COL_r, PROP_mes, DONE_mes, IDLE_a, COL_a, PROP_a, DONE_a, and $\mathcal{A}$ that do not assign to the underlying variables of Π. As will be shown, this significantly reduces the proof effort for specific applications of our base algorithms. The proof of specification (1) can be found in [32].

8 Propagation of Information with Feedback

PIF (**P**ropagation of **I**nformation with **F**eedback), is the problem of broadcasting a piece of information I to all nodes in a connected network in such a way that all the nodes "know" when they have finished participating in the broadcast, one node in particular (the *starter*) "knows" that the broadcast is completed, and upon completion all nodes own the piece of information I [25]. A node p "knows" that it has finished participating in the broadcast when it has received and sent messages from and to all its neighbors, i.e. when *done.p*. Moreover, when the *starter* is *done* it "knows" that the broadcast is completed. To make Π suitable for the PIF application, we introduce new local variables $V.p$ for all processes $p \in \mathbb{P}$ and instantiate Π such that: (a) the starter initially has I stored in $V.starter$ **;** (b) upon receipt of a message m in the IDLE phase of node p, this value is copied directly to $V.p$ **;** (c) in the PROP phase of node p, the value stored in $(V.p)$ is sent **;** (d) the contents of the messages received in the col phase are discarded **;** (e) the contents of the messages sent in the DONE phase can remain unspecified. Now PIF can be defined as:

$\text{PIF}_\Pi = \Pi.initA.\text{IDLE\_r}.\text{COL\_r}.\text{PROP\_mes}.\text{DONE\_mes}.\text{IDLE\_a}.\text{COL\_a}.\text{PROP\_a}.\text{DONE\_a}.\mathcal{A}.\mathcal{V}$

where DONE_mes is arbitrary, IDLE_a = COL_a = PROP_a = DONE_a = skip, $\mathcal{A} = \emptyset$, $\mathcal{V} = \{p \in \mathbb{P} \mid V.p\}$, and:

$initA \ = \ (\lambda s \cdot ((s \circ V).starter) = I)$
IDLE_r$= (\lambda p, q, m \cdot V.p := m)$
COL_r$= (\lambda p, q, m \cdot \mathsf{skip})$
PROP_mes$= (\lambda p, q \cdot V.p)$

The formal specification of PIF applications reads:

$$(J_\Pi \wedge J_{\text{PIF}}) \ {}_{\text{PIF}\_\Pi}\vdash \mathbf{ini}\text{PIF}_\Pi \rightsquigarrow (\forall p : p \in \mathbb{P} : done.p) \ \wedge \ (\forall p : p \in \mathbb{P} : (V.p) = I) \tag{2}$$

where J_{PIF} is an invariant of PIF_Π stating additional safety behavior. Since PIF_Π is an instantiation of Π, the correctness criterion above can be reduced:
(2)
$\Leftarrow$ (Thm.3)
$(J_\Pi \wedge J_{\text{PIF}}) \ {}_{\text{PIF}_\Pi}\vdash \mathbf{ini}\text{PIF}_\Pi \rightsquigarrow (\forall p : p \in \mathbb{P} : done.p)$
$(J_\Pi \wedge J_{\text{PIF}}) \ {}_{\text{PIF}_\Pi}\vdash \forall p : p \in \mathbb{P} : done.p \rightsquigarrow \forall p : p \in \mathbb{P} : (V.p) = I$

The first conjunct is implied by specification (1) (use Thm.1, Thm.2, and the fact that $\mathbf{ini}\text{PIF}_\Pi$ includes $\mathbf{ini}\Pi$). The second conjunct is application specific and, using Thm.4 and Thm.6, can be reduced to:
${}_{\text{PIF}_\Pi}\vdash \circlearrowleft (J_\Pi \wedge J_{\text{PIF}})$
$\forall p : p \in \mathbb{P} : (J_\Pi \wedge J_{\text{PIF}} \wedge done.p \Rightarrow (V.p) = I)$
The only thing left to do is find the characterization of the invariant J_{PIF} that establishes the last conjunct. The most straightforward and evident choice is: $(\forall p : p \in \mathbb{P} : done.p \Rightarrow (V.p) = I)$. This is obviously a stable property, and sufficient to prove the conditions stated above.

9 Computation of Summation

Suppose that every node $p \in \mathbb{P}$ in the network has a unique local variable that stores some data value, and that the distribution of these local variables is given by a function $V : \mathbb{P} \rightarrow \texttt{Var}$. Imagine $D : \mathbb{P} \rightarrow \texttt{Val}$ being a snapshot of the values that reside at the local variables in some state s, i.e. $D = s \circ V$.

It is straightforward [8, 17, 18, 29, 30] to instantiate our algorithm such that the sum of all data values is computed and eventually resides at the *starter* (i.e. is stored in the variable $V.starter$). Instantiate Π such that: D_i is defined to be the initial distribution of the data values**;** in the (PROP) phase of node p, the identity element of $+$ (say 0) is sent**;** in the (DONE) phase of node p, $(V.p)$ is sent**;** upon receiving a message m in the (IDLE) and (COL) phase, this value m is added to the data value that resides in $(V.p)$, and the result is stored in $(V.p)$. We define the specific SUM-application of Π by

$$\mathrm{S}_\Pi = \Pi.initA.\text{IDLE\_r}.\text{COL\_r}.\text{PROP\_mes}.\text{DONE\_mes}.\text{IDLE\_a}.\text{COL\_a}.\text{PROP\_a}.\text{DONE\_a}.\mathcal{A}.\mathcal{V}$$

where IDLE_a=COL_a=PROP_a=DONE_a=skip, $\mathcal{A} = \emptyset$, $\mathcal{V} = \{p \in \mathbb{P} \mid V.p\}$ and, for a commutative monoid $(+, 0)$:

$initA = (\lambda s \cdot D_i = (s \circ V))$
$\text{IDLE\_r} = \text{COL\_r} = (\lambda p, q, m \cdot V.p := V.p \ + \ m)$
$\text{PROP\_mes} = (\lambda p, q \cdot 0)$
$\text{DONE\_mes} = (\lambda p \cdot V.p)$

The formal specification of these summation algorithms reads:

$$J_\Pi \wedge J_S \ {}_{\mathrm{S}_\Pi}\vdash \ \mathbf{ini}\mathrm{S}_\Pi \rightsquigarrow (V.starter) = \sum_{x \in \mathbb{P}} D_i.x \tag{3}$$

where J_S is an invariant stating additional safety behavior of S_Π. We now reduce the correctness criterion for S_Π in a way that progress properties already proved for Π are inherited and we only need to prove application specific properties.

(3)
$\Leftarrow$ (Thm.1, Thm.2, $\mathbf{ini}\mathrm{S}_\Pi$ includes $\mathbf{ini}\Pi$)
$J_\Pi \ {}_{\mathrm{S}_\Pi}\vdash \ \mathbf{ini}\Pi \rightsquigarrow (\forall p : p \in \mathbb{P} : done.p)$
${}_{\mathrm{S}_\Pi}\vdash \circlearrowleft (J_\Pi \wedge J_S)$
$J_\Pi \wedge J_S \wedge (\forall p : p \in \mathbb{P} : done.p) \Rightarrow \ V.starter = \sum_{x \in \mathbb{P}} D_i.x$

The first conjunct equals (1). Consequently, we are left with finding the characterization of invariant J_S that establishes the last conjunct. Although some ingenuity is required in order to come up with this invariant, its construction is guided by the availability of information about its use within the process of verifying the above correctness criterion. Any message sent during the execution of the algorithm is: either 0 (in the PROP phases), or the data value $(V.p)$ residing at some node p (in the DONE phases). However, after sending $(V.p)$, node p will be *done*. Hence, since 0 is the identity element of $+$, the desired sum SUM will, in any state, be: the sum of values that reside at the nodes that are not *done*, added to the sum of values that are in transit in s. The safety property J_S in state s:

$$\begin{aligned}\textstyle\sum_{p \in \mathbb{P}} D_i.p \ = & (s \circ V).starter \\ & + \textstyle\sum_{p \in \mathbb{P} \wedge (p \neq starter) \wedge \neg done.p} (s \circ V).p \\ & + (\text{the values in the communication channels in } s)\end{aligned}$$

With this definition we can prove the required application specific requirements from above. Note that invariant J_Π is still unspecified at this point, and its precise characterization is not needed to be able to derive a proof strategy for the summation application. Consequently, any instantiation of the DS, ECHO, PIF, TARRY and DFS algorithms that maintains safety property J_S can be used to compute the sum. This clear separation of progress and safety properties enables us to prove the correctness of any application by verifying the safety property of the application and inheriting the progress proof (including invariant J_Π that was needed to prove this progress) of the underlying algorithm.

10 Mattern's Four Counter Solution for Termination Detection

The problem of termination detection is to superimpose a *control algorithm* on a given so-called *basic computation* such that the first can detect when the termination condition holds for the latter.

In [20] Mattern proposes a termination detection solution called the Four Counter Method (FCS). The solution departs from a basic computation with atomic actions, arbitrary message delays and processes (taken from a set $\mathbb{P}$) that are always "passive"", meaning that a process may at any time take any message from one of its incoming communication channels, change it its local state and send out any number of messages. Such a distributed computation is considered to be *terminated* if there are no messages in transit. In order to be able to test this condition, every process $p \in \mathbb{P}$ of the basic computation keeps a counter $s.p$ for the number of sent basic messages, and a counter $r.p$ for the number of received basic messages. The control algorithm visits all the processes during a first (red) round to accumulate the counters by $\mathcal{S} = \sum_{p\in\mathbb{P}}\ s.p$ and $\mathcal{R} = \sum_{p\in\mathbb{P}}\ r.p$. Then a second (green) round is started after completion of the first, yielding the accumulated counter values $\mathcal{S}'$ and $\mathcal{R}'$. In [20] it is proved that the termination condition (i.e. there are no messages in transit) holds if $\mathcal{S} = \mathcal{R} = \mathcal{S}' = \mathcal{R}'$.

The two rounds of this control algorithm can easily be implemented by any diffusing computation. The red round is when all processes get non-idle, the green round is when all processes become done. In order to capture the current values of $s.p$ and $r.p$ at both rounds, we introduce four variables for each process $p \in \mathbb{P}$ in the control algorithm: $s_{red}.p$, $s_{green}.p$, $r_{red}.p$, $r_{green}.p$. Moreover, we assign $s_{red}.p$, $r_{red}.p := s.p$, $r.p$ when p becomes non-idle, and $s_{green}.p$, $r_{green}.p := s.p$, $r.p$ when p becomes done. To accumulate the counter values, we introduce, for each process $p \in \mathbb{P}$, four variables $S.p$, $S'.p$, $R.p$, and $R'.p$ that respectively store the intermediate sum of the $s_{red}.q$, $s_{green}.q$, $r_{red}.q$, and $r_{green}.q$ values of those processes q that reside in the subtree that is rooted at p. Consequently, the rooted spanning tree that is created by the diffusing computation guarantees that, at the end of the control algorithm when the *starter* is done, it holds that:

$\mathcal{S} = \sum_{p\in\mathbb{P}}\ s_{red}.p = S.starter\ + s_{red}.starter$,
$\mathcal{R} = \sum_{p\in\mathbb{P}}\ r_{red}.p = R.starter\ + r_{red}.starter$,
$\mathcal{S}' = \sum_{p\in\mathbb{P}}\ s_{green}.p = S'.starter\ + s_{green}.starter$,
$\mathcal{R}' = \sum_{p\in\mathbb{P}}\ r_{green}.p = R'.starter\ + r_{green}.starter$,

and hence the *starter*, when it is done, can determine whether the termination condition holds. For this we will introduce a *new* variable term, that will be assigned by the *starter* when its done.

In the previous algorithms and their applications, the *starter* spontaneously started its execution by becoming non-idle. In the case of the FCS the starter should not only become non-idle, but also initialize $s_{red}.starter$ and $r_{red}.starter$ to $s.starter$ and $r.starter$ respectively. As before, we reflect this within the initial condition (**init**) of the algorithm. Consequently, we specify de algorithms as follows:

$\mathrm{FCS}_\Pi = \Pi.initA.\textsc{idle\_r}.\textsc{col\_r}.\textsc{prop\_mes}.\textsc{done\_mes}.\textsc{idle\_a}.\textsc{col\_a}.\textsc{prop\_a}.\textsc{done\_a}.\mathcal{A}.\mathcal{V}$

where $\textsc{idle\_r} = \textsc{col\_a} = \textsc{prop\_a} = \mathsf{skip}$, and:

$initA = \forall p \in \mathbb{P} : S.p = R.p = S'.p = R'.p = 0 \wedge s_{red}.starter = s.starter\ \wedge\ r_{red}.starter = r.starter$
$\textsc{idle\_a} = s_{red}.p, r_{red}.p := s.p, r.p$
$\textsc{col\_r} = \lambda p, q, m \cdot$ if $m =$<green $s\ s'\ r\ r'$>
then $S.p, R.p := S.p + s, R.p + r \parallel S'.p, R'.p := S'.p + s', R'.p + r'$
$\textsc{prop\_mes} = \lambda p, q \cdot$ <red>
$\textsc{done\_mes} = \lambda p, q \cdot$ <green $(S.p + s_{red}.p)\ (S'.p + s.p)\ (R.p + r_{red}.p)\ (R'.p + r.p)$>
$\textsc{done\_a} = s_{green}.p, r_{green}.p := s.p, r.p$
$\mathcal{A} = \{$ if $done.starter$
then term $:= (S.starter\ + s_{red}.starter) = (S'.starter\ + s.starter)$
$= (R.starter\ + r_{red}.starter) = (R'.starter\ + r.starter)$
$\parallel s_{green}.starter := s.starter \parallel r_{green}.starter := r.starter\ \}$
$\mathcal{V} = \{\{s_{red}.p, s_{green}.p, r_{red}.p, r_{green}.p\} \mid p \in \mathbb{P}\} \cup \{\{S.p, S'.p, R.p, R'.p\} \mid p \in \mathbb{P}\} \cup \{\mathsf{term}\}$

Note that in DONE_mes and $\mathcal{A}$ we use the s and r values instead of s_{green} and r_{green}. We have to do this because UNITY does not have sequential composition of assignment statements. However, since the assignment statements are executed in parallel, the result is the same.

The formal specification of these algorithms reads:

$$J_\Pi \wedge J_{\textsc{fcs}\ \mathrm{FCS}_\Pi} \vdash \mathbf{ini}\mathrm{FCS}_\Pi \rightsquigarrow \mathsf{term} \equiv \sum_{p\in\mathbb{P}} s_{red}.p = \sum_{p\in\mathbb{P}} s_{green}.p = \sum_{p\in\mathbb{P}} r_{red}.p = \sum_{p\in\mathbb{P}} r_{green}.p \qquad (4)$$

where $J_{\textsc{fcs}}$ is an invariant stating additional safety behavior of FCS_Π. Again, we can reduce the correctness criterion for in a way that progress properties already proved for Π are inherited and we only need to prove application specific properties.

(4)
$\Leftarrow$ (Thm.5, Thm.1, Thm.2, $\mathbf{ini}\mathrm{FCS}_\Pi$ includes $\mathbf{ini}\Pi$)

$J_{\Pi\ \mathrm{FCS}\Pi}\vdash\ \mathbf{ini}\Pi \rightsquigarrow (\forall p : p \in \mathbb{P} : \mathit{done}.p)$
$_{\mathrm{FCS}\Pi}\vdash\circlearrowright (J_\Pi \wedge J_{\mathrm{FCS}})$
$J_\Pi \wedge J_{\mathrm{FCS}\ \mathrm{FCS}\Pi}\vdash (\forall p : p \in \mathbb{P} : \mathit{done}.p) \rightsquigarrow \mathsf{term} \equiv \sum_{p\in\mathbb{P}} s_{red}.p = \sum_{p\in\mathbb{P}} s_{green}.p = \sum_{p\in\mathbb{P}} r_{red}.p = \sum_{p\in\mathbb{P}} r_{green}.p$

Again, the first conjunct can be proved by (1), since the new assignments superimposed on Π do not write to Π's underlying variables. The last conjunct, using Thm.4, can be refined to

$$J_\Pi \wedge J_{\mathrm{FCS}} \wedge (\forall p : p \in \mathbb{P} : \mathit{done}.p)\ \mathsf{ensures}\ \mathsf{term} \equiv \sum_{p\in\mathbb{P}} s_{red}.p = \sum_{p\in\mathbb{P}} s_{green}.p = \sum_{p\in\mathbb{P}} r_{red}.p = \sum_{p\in\mathbb{P}} r_{green}.p$$

This one-step progress properties is satisfied by the action in $\mathcal{A}$, and so we are left with finding the characterization of invariant J_{FCS} that enables us to prove this. Since four summations are being calculated simultaneously, we will have four invariants that are similar to the invariant of the summation algorithm from the previous section. More specifically, J_{FCS} will be defined for all quadruples $(X, c, x, n) \in \{(S, red, s, 1), (S', green, s, 2), (R, red, r, 3), (R', green, r, 4)\}$ as:

$$\begin{aligned}\textstyle\sum_{p\in\mathbb{P}} x_c.p = {} & X.starter + x_c.starter \\ & + \textstyle\sum_{p\in\mathbb{P}\wedge(p\neq starter)\wedge\neg done.p} X.p + x_c.p \\ & + (\text{the } n\text{th-value of the green messages in transit})\end{aligned}$$

With this definition of J_{FCS}, we can prove the required application specific requirements from above.

11 Conclusion

We have given a highly parameterized abstract algorithm for diffusing computations, and we have shown that many existing algorithms and their applications can be obtained from this abstract algorithm by instantiating the parameters appropriately. By clearly separating the progress and safety properties of the different applications, we can proof the correctness of any application by verifying an application-specific safety property and inheriting the already proved termination and safety of the underlying algorithm.

Studies with postgraduate students have given empirical evidence that our re-usable specifications and proofs improve their understanding of the algorithms, increase the slope of the learning-curve, and enable students to prove the correctness of some specific applications within a remarkably short amount of time.

Theorems About $\circlearrowright$ and $\rightsquigarrow$

Thm 1 $\rightsquigarrow$ *Substitution:* $\dfrac{(J \wedge p \Rightarrow q) \wedge\ J\ _P\vdash (q \rightsquigarrow r) \wedge (J \wedge r \Rightarrow s)}{J\ _P\vdash p \rightsquigarrow s}$

Thm 2 $\rightsquigarrow$ *Stable Strengthening:* $\dfrac{(\ _P\vdash\circlearrowright (J_1 \wedge J_2))\ \wedge\ J_1\ _P\vdash p \rightsquigarrow q}{(J_1 \wedge J_2)\ _P\vdash p \rightsquigarrow q}$

Thm 3 $\rightsquigarrow$ *Accumulation:* $\dfrac{(J\ _P\vdash p \rightsquigarrow q)\ \wedge\ (J\ _P\vdash q \rightsquigarrow r)}{J\ _P\vdash p \rightsquigarrow q \wedge r}$

Thm 4 $\rightsquigarrow$ *Introduction:* $\dfrac{\circlearrowright (J \wedge q) \wedge ([J \wedge p \Rightarrow q]\ \vee\ (p \wedge J\ \mathsf{ensures}\ q))}{J\ _P\vdash p \rightsquigarrow q}$

Thm 5 $\rightsquigarrow$ *Trans:* $\dfrac{J\ _P\vdash (p \rightsquigarrow q)\ \wedge\ J\ _P\vdash (q \rightsquigarrow r)}{J\ _P\vdash p \rightsquigarrow r}$

Thm 6 $\rightsquigarrow$ *Conjunction:* $(W \neq \emptyset)$ $\dfrac{(\forall i : i \in W : J\ _P\vdash p.i \rightsquigarrow q.i)}{J\ _P\vdash (\forall i : i \in W : p.i) \rightsquigarrow (\forall i : i \in W : q.i)}$

References

[1] A. Arora and M.G. Gouda. "Distributed Reset" *IEEE Trans. on Comp.*, 43(9):1026–1038, 1994.

[2] K.M. Chandy and J. Misra. "Distributed computations on graphs" *Com. ACM*, 25(11):833–838, 1982.

[3] K.M. Chandy and J. Misra, "Termination detection of diffusing computations in communicating sequential processes" *ACM Tr. Prog. Lang. and Sys.* 4(1):37–43, 1982.

[4] K.M. Chandy and J. Misra. "Distributed deadlock detection." *ACM Tr. Comp. Sys.* 1(2):144–156, 1983.

[5] K.M. Chandy and J. Misra. *Parallel Program Design – A Foundation*, AW, 1988.

[6] E.J.H. Chang. "Echo algorithms" *IEEE Trans. Softw. Eng.*, (SE-8):391–401, 1982.

[7] T.-Y. Cheung. "Graph traversal techniques and the maximum flow problem in distributed computation" *IEEE Tr. on Softw. Eng.*, SE-9(4):504–512, 1983.

[8] C.-T. Chou. "Mechanical verification of distributed algorithms in higher order logic" *Proc. 7th Workshop on HOLTP*, T. Melham and J. Camilleri (eds) *LNCS* 859, 1994.

[9] C.-T. Chou. "Using operational intuition about events and causality in assertional proofs." TR 950013, UCLA, Feb. 1995.

[10] A. Cournier, A. Datta, F. Petit, and V. Villain. "Self-Stabilizing PIF algorithm in arbitrary rooted networks." *21th ICDCS*, pages 91–98, 2001.

[11] E.W. Dijkstra and C.S. Scholten. "Termination detection for diffusing computations." *Information Processing Letters*, 11(1):1–4, 1980.

[12] S.G. Finn. "Resynch procedures and a fail-safe network protocol." *IEEE Trans.*, COM-27:840–845, 1979.

[13] M. Filali, P. Mauran, G. Padiou, P. Quinnec, X. Thirioux. "Refinement based validation of an algorithm for detecting distributed termination." *FMPPTA2000*, LNCS 1800, 2000.

[14] N. Francez. "Distributed termination" *ACM Trans. Prog. Lang.*, 2(1):42–55, 1980.

[15] J.J. Garcia-Lunes-Aceves. "Loop-free routing using diffusing computations." *IEEE/ACM Trans. on Netw.*, 1(1):130–141, 1993.

[16] M.J.C. Gordon and T.F. Melham. *Introduction to HOL.* CUP, 1993.

[17] J. Groote, F. Monin, and J. Springintveld. "A computer checked algebraic verification of a distributed summation algorithm." TR-CSR-97-14, TUE, 1997.

[18] W.H. Hesselink. "A mechanical proof of Segall's PIF algorithm." TR-CS-R9604, UG, 1996.

[19] S. Huang. "A Distributed Deadlock Detection Algorithm for CSP-Like Communication," *ACM Tr. on Prog. Lang. and Sys*, 12(1):102–122, 1990.

[20] F. Mattern. "Algorithms for Distributed Termination Detection." *Distributed Computing*, vol 2, pp 161–175, 1987.

[21] F. Mattern. "Distributed Control Algorithms (Selected Topics)." *Parallel Computing on Distributed Memory Multiprocessors*, F. Özguner, F. Ercal (Eds.), pp. 167–185, 1993.

[22] L. Onana Alima. *Self.stabilization by self-stabilizing waves.* UCL 2000.

[23] W. Prasetya. *Mechanically Supported Design of Self-stabilizing Algorithms.* UU 1995.

[24] M. Raynal and J-M. Helary. *Synchronisation and control of distributed systems.* 1990.

[25] A. Segall. "Distributed network protocols." *IEEE Trans. IT*, (IT-29):23–35, 1983.

[26] F.A. Stomp. *Design and verification of Distributed Network Algorithms: Foundations and Applications.* PhD thesis, TUE, 1989.

[27] M. G. Tarry. "Le problème des labyrinthes." *Nouvelles Annales de Mathématique*, (149):187–190, 1895.

[28] G. Tel. *The Structure of Distributed Algorithms.* PhD thesis, UU, 1989.

[29] G. Tel. *Introduction to Distributed Algorithms.* CUP, 1994.

[30] F.W. Vaandrager. Verification of a distributed summation algorithm. CS-R9505, CWI 1995

[31] T.E.J. Vos and S.D. Swierstra. "Program refinement in UNITY" UU-CS-2001-41, 2001.

[32] T.E.J. Vos and S.D. Swierstra. "Proving distributed hylomorphisms" UU-CS-2001-40, 2001.

Process Modeling Architectures with Namespace and XML Technology

Tiago Lopes Telecken, José Valdeni de Lima
Universidade Federal do Rio Grande do Sul, Instituto de Informática
Porto Alegre – RS, Brazil, Av. Bento Conçalves, 9500 Campus do Vale - Bloco IV
CEP 91591-970, Caixa Postal 15064
{telecken, valdeni} @inf.ufrgs.br

Montgomery Barroso França
Universidade Federal do Rio Grande do Sul, Instituto de Informática

Banco Central do Brasil
Brasília – DF, Brazil, CEP 70074-900, SBS Quadra 3 Bloco B - Ed. Sede
Caixa Postal 08670
mont@bcb.gov.br

Abstract

The necessity of productivity and quality in workflow systems demands the use of several process modeling architectures. However, in the workflow area, there is few information about optional architectures of relationships among models and documents used in the process modeling phase. To attend the demand for information about optional architectures, this paper presents a survey about *many-one* architecture and a comparative study about process modeling architectures. The *many-one* architecture uses namespace and XML technology to insert elements of many XML models in only one process definition. Such characteristic allows a workflow technology development be more modular and reusable.

Keywords: Process Definitions, Process modeling, Namespace, Workflow, XML

1 Introduction

In the document structuring area the most traditional architecture of relationships among models and documents is formed by one document that contains all information used in an application and by one model that defines all structure of referred document. This is the *one-one* architecture.

But, some applications need architectures that contain many models. Two optional architectures were developed for these applications. The first one is formed by many documents. Its respective model defines the structure of each document. This is the *many-many* architecture. The second one is formed by one document, but the structure of this document is defined in many models. This is the *many-one* architecture.

The necessity of productivity and quality in workflow systems and the application of workflow technology in more and more complex environments demand the use of all process modeling architectures.

However, there are few information, developments and researches about architectures of relationships among models and documents used in the process modeling phase. And, this few information is concentrated in the *one-one* architecture. For helping developers and researches to choose the best architecture in process definitions, this paper presents a survey about *many-one* architecture and a comparative study about the process modeling architectures.

2 Overview -Workflow Systems Architecture

In agreement with the Workflow Management Coalition (WfMC) [20][18], the generic architecture of a workflow system should follow the model shown in the figure 1. Concisely this model determines that: the workflow designers can generate a process definition through a definition tool. The process definition should contain (i) all the information that the workflow engine needs to manage, control and execute a workflow; and (ii) all the information that the definition tool needs to facilitate the process definition edition. After, the process definition can be sent to a workflow engine. The workflow engine will interpret the process definition. After, it will control, manage and execute the workflow described in the process definition.

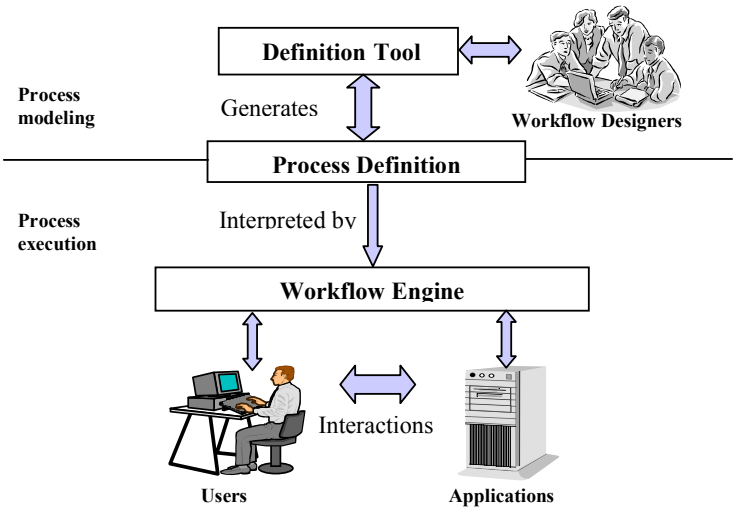


Fig. 1. Workflow Systems

The elements and attributes contained in a process definition are defined in one model called *process definition model*. The process definitions are of two types. The first one is the internal representations. Internal representations were projected to be the internal data representation of a specific definition tool. The second one is the interchange patterns. The interchange patterns allow the compatibility among the systems evolved in a workflow process.

Process definitions are complex documents that have a rigid syntax. So, in the process modeling phase, the use of a definition tool is essential. The referred tool facilitates the analysis, modeling and codification of a process definition.

The main components and functionalities of the definition tool are [15]:

- **Internal representation:** It is a process definition which structure is defined by the process definition model adopted by the definition tool.
- **Internal representation control**: It is the internal representation edition resources.
- **Visions**: They are functionalities that supply the users with one vision of the internal representation. Graphic and textual visions can be supplied.

- **Export/Import process definitions**: Export is the resource that converts the internal representation of a workflow process to an external format. Import is the resource that converts a process definition of an external format to the internal representation.
- **Nesting**: They are functionalities that allow the process definition hierarchy modeling and navigation.
- **Error verification**: It is a functionality that allows the automatic error verification in process definitions. Syntax and semantic errors are verified.
- **Analysis and simulation:** They are functionalities that facilitate the analysis and simulation of modeled workflow behavior.

3 Overview - Using XML in Process Definitions

Extensible Markup Language (XML) technology [4] is a vast and growing set of modules that offer services, tools and standards used in a wide range of areas. Its use in document structuring is largely divulged and offers a growing number of tools.

XML document structuring is largely used in process definition. The XML Process Definition Language (XPDL) [19] is an important XML interchange pattern defined by WfMC (entity created for developing workflow standards). Other important XML interchange patterns are [1]: XLang [17] from Microsoft, Web Services Flow Language (WSFL) [11] from IBM, Business Process Modeling Language (BPML) from Business Process Modeling Language Initiative (BPMI) [2], Web Service Choreography Interface (WSCI) [23] from Sun/BEA and Business Process Execution Language for Web Services (BPEL4WS) [12] (the evolution of XLang and WSFL in the web services context).

Many internal representations of process definitions are XML languages. This is the case of Biztalk Orchestration Designer that uses the XLang [17] and of IBM's MQ Series Workflow that has an internal representation based on WSFL [9].

Most definition tools export and/or import their internal representations for one or several XML formats. The most common exportations and importations are for the following formats: XPDL, XLang, WSFL, BPML, WSCI and BPEL4WS.

The main advantages of XML application in process definitions are:

- A great interoperability with workflow systems (workflow management systems, definition tools, simulation programs, etc). The most important workflow systems are enabled to import, export or interpret XML languages.
- A great interoperability with systems from other areas. XML documents are a standard data exchange format. With XML the process definition and the definition tools become compatible with many tools, protocols, applications and resources such as Extensible Stylesheet Language Transformations (XSLT), XML Schema, Simple API for XML (SAX), Document Object Model (DOM) API, and much more.

The use of XML in process definitions is growing. The main entities and organizations of workflow area already use XML representations. For this reason, the focus of this paper is XML solutions.

3.1 Using Namespace in Process Definition

The specification Namespace [3] is a XML standard that defines how two or more XML representations can be inserted in the same document. This specification is an official recommendation of the World Wide Web Consortium (W3C).

The namespace is used to insert external XML representation into process definitions. This is the case of Resource Description Framework (RDF) elements and attributes inserted in PSL [14]. RDF elements in Process Specification Language (PSL) documents define resources used in workflow systems.

The namespace is also used to put elements of a process definition language in external XML documents. This is the case of XLang elements that are put in Web Services Description Language (WSDL) documents. According to Thatte [17], the WSDL is a net service description protocol. Among other services, the WSDL describes Web Services by XLang elements and attributes.

3.2 Using SVG, XLink, and RDF in Process Definition

The Scalable Vector Graphics (SVG) [22], XML Linking Language (XLink) [6] and RDF [21] specifications are XML vocabularies that describe respectively 2D graphics, links and resources. These languages are official recommendations of World Wide Web Consortium (W3C) and are cited in the cases of this paper. Definition tools such as ILOG [10] export the graphic representation of workflow process to SVG formats. The XLink is used by some definition tools and is proposed by some authors [7] as a good standard to link elements of process definitions localized in different documents.

4 Related works - The Process Modeling Architectures

The approached area of this paper is the architectures of relationships among models and documents used in the process modeling phase. In these architectures, the entities that are relevant for process execution or process modeling are defined in models. The models are applied in domain areas. The entities of a domain that are defined in one model are called entity domains. The entity domains are represented in documents by elements and attributes. The structure of documents used in these architectures is defined in the referred models. There are three possible architectures.

In the first architecture, each process definition is formed by one document. The structure of process definition is defined in only one model. This is the *one-one* architecture. In the current process definition and definition tools, this is the most used architecture. An example of this architecture is XPDL [19]. In this standard, all elements used in the process modeling phase and in the process execution phase are defined in one model and represented in one document.

In the second architecture, each process definition is formed by more than one document. The structure of each document is defined in one different model. The elements of each document may make mutual references. So, the documents that form one process definition need resources to maintain the mutual synchronization. This is the *many-many* architecture. An example is the internal representation of definition tool FORO process designer [8], that is formed by two documents. One document contains elements defined in the process model. The other contains elements defined in the informational model. There are two models and the process definition is formed by two documents.

In the third architecture, each process definition is formed by one document. And the structure of process definition is defined in different models. This is the *many-one* architecture. An example is the PSL [14], an interchange pattern that defines external resources by RDF elements inserted by namespace standard. There are two models, PSL and RDF. The PSL is the main model and RDF is the secondary model.

Each column of table 1 shows the characteristics of one researched architecture and the figure 2 shows the architecture of the three referred examples.

Table 1 - Architecture's characteristics

1- *One-One*	2- *Many-Many*	3- *Many-One*
Process definition structure defined in **one model**	Process definition structure defined in **many models**	Process definition structure defined in **many models**
Process definition is formed by **one document**	Process definition is formed by **many documents**	Process definition is formed by **one document**

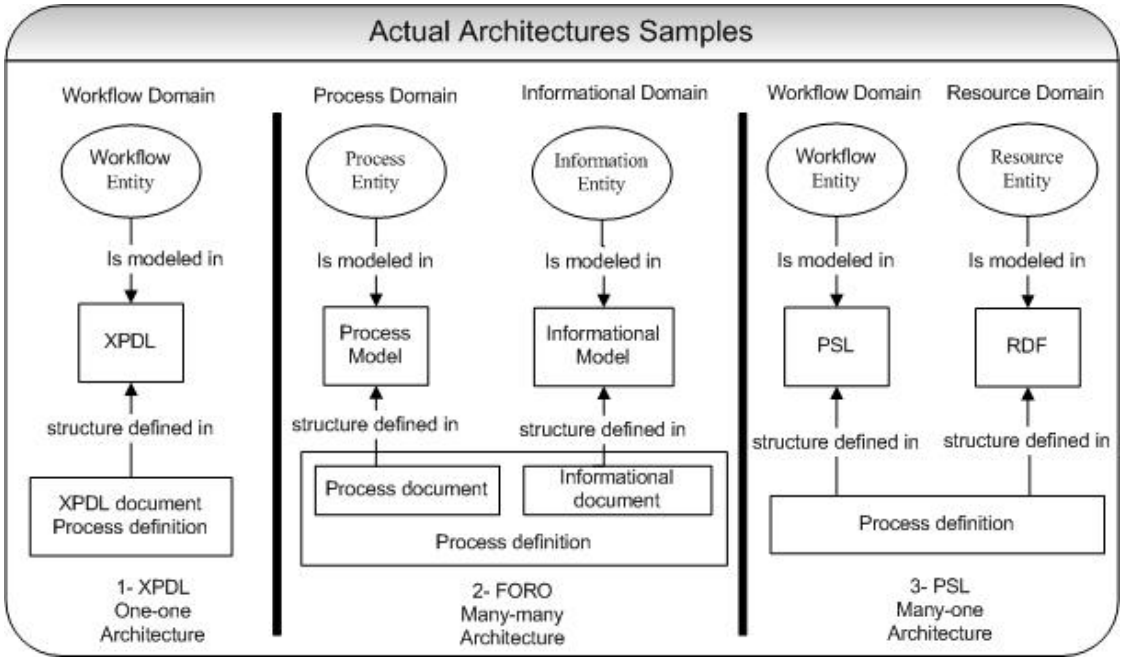


Fig. 2. Architecture's samples

4.1 XML Implementation

For implementing the process modeling architectures with XML technology it is necessary: (i) to define the models with a document type definition (DTD) or a XML Schema; and (ii) to implement applications for interpretation and manipulation of XML files.

In the *one-one* architecture, the process definition is a XML file that is in conformance with the only defined model.

In the *many-many* architecture, each document of a process definition is a XML file that is in conformance with the correspondent defined model. Another important implementation in this kind of architecture is the implementation of resources that make the document synchronizations. The document synchronizations are necessary for relationship maintenance that exists between elements of different documents.

In the *many-one* architecture, the definition of the integration rules of involved models is needed. An integration model can make the integration rules. To define the integration model in XML, a DTD or a XML Schema can be used. In the integration model, one of the integrated models is the main model and the others are the secondary models. The main model elements can be inserted normally in the integration DTD and the elements of the other models are inserted by namespace's standard (Generally, the integration model is an adaptation of main model). In the *many-one* architecture, the process definition is a XML file that is in conformance with the integration model. The figure 3 shows a more detailed *many-one* architecture.

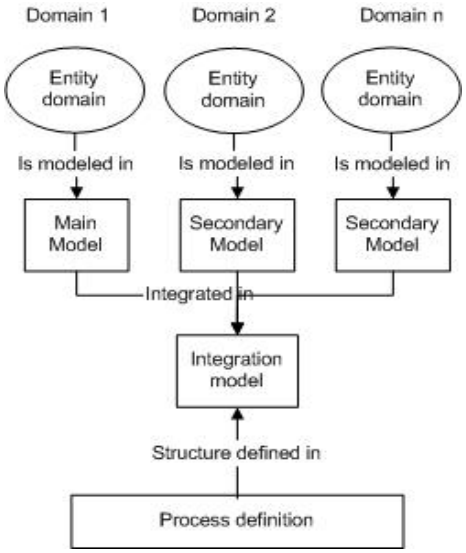


Fig. 3. Detailed *many-one* architecture

5 *Many-One* Case Study I - The Amaya Workflow Prototype

This *many-one* case study is presented to show more details about the *many-one* architecture application. The Amaya Workflow (AW)[16] [13] is a definition tool developed in Universidade Federal do Rio Grande do Sul (UFRGS). The AW was developed as an extension of Amaya [24] XML. The Amaya is a XML editor and browser, developed in Institut National de Recherche en Informatique et en Automatique (INRIA). The internal representation model of AW is similar to the model proposed by Casati et al. [5]. Following, some elements of AW model are described:

Workflow: It represents a process. It is the root element.
Task: It represents a workflow task.
Connector: It represents the connections between workflow elements.
MultiActivity and **SuperActivity**: They represent tasks that can be expanded.
The other elements are different types of joins and forks.

The internal representation of AW includes some SVG and Xlink attributes in the AW elements. In the beginning of internal representation, the namespaces of XLink, SVG and AW model need to be defined. The AW is the main model. XLink and SVG are secondary models. An example is shown following:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE AW PUBLIC http://www.inf.ufrgs/~telecken/AW/AW.dtd">
```

<workflow xmlns=" http://www.inf.ufrgs/~telecken/AW/"
xmlns:svg="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" version="1.0">

5.1 The Graphic Vision

Associated with each element of AW there is one graphic representation used in one AW graphic vision. The graphic representation is equivalent to the representation proposed in WIDE project [5]. The AW graphic vision is a functionality that helps the workflow designers in the workflow edition and workflow analysis.

The SVG attributes are used for storing of entities about the workflow graphic appearance. Each AW element contains SVG attributes that describe the correspondent graphic representation. For example, into the Task element of AW, the stroke, fill, x, y, width, Stroke-width and height attributes of element rect need to be inserted. The rect is an SVG element. To inform that these attributes belong to rect element, the type attribute with the value "rect" needs to be inserted. The type attribute is not a SVG attribute but it is used in this research. The table 2 shows what each inserted attribute represents. Each column contains information about the attribute informed in the table header. All attributes describe a rectangle.

Table 2 - The attributes of *rect*.

Stroke	stroke-width	fill	X	Y	width	height
Stroke color	Stroke width	Internal color	Coordinate X of left superior corner	Coordinate Y of left superior corner	Width	Height

An example of *Task* element is shown following.

<Task name="Fill document" ID="1234" application="WebForm" description="The manager fill the document"
svg:type=”rect” svg:stroke="black" svg:fill="white" svg:y="31px" svg:x="40px" svg:width="90px" svg:height="40px"
svg:stroke-width="2"/>

The SVG attributes are prefixed by "svg:". The other attributes are workflow attributes of AW model. The same procedure is applied on the other elements until all workflow graphic representation is defined by SVG attributes.

5.2 The Nesting

The AW has nesting functionality. The *SuperTask* and *MultiTask* elements aim to external process definitions. The process definition aimed by *SuperTask* or *MultiTask* describes the activities of these elements. The link of these documents is made by XLink attributes. Into the *MultiTask* or *SuperTask* elements of AW model, the type and href attributes of XLink simple element need to be inserted. To inform that these attributes belong to simple element, the type attribute with the value "simple" need to be filled. The value of href is a valid Uniform Resource Locator (URL) that aims to one process definition. An example of *SuperTask* element is shown following:

<SuperTask name="Fill document" ID="1234" application="WebForm" description="The manager fill the document"
svg:type=”rect” svg:stroke="black" svg:fill="white" svg:y="31px" svg:x="40px" svg:width="90px" svg:height="40px"
svg:stroke-width="10" xlink:type= “simple” xlink:href=”./exec.xml”/>

The two XLink attributes are prefixed by "xlink:".

5.3 Other Functionalities and the Namespace

Other functionalities of AW are: error verification; exportation to the XPDL, SVG and PDF formats; textual visions synchronized with graphic visions; internal representation controls; cooperation resources; etc. All AW functionalities use XML resources and the process definitions. The use of namespace standard does not disturb these functionalities. Basically, an attribute added by namespace is accessed in the same way as other attributes. The difference is the name of the attributes. The namespace attributes are prefixed by the model name and a colon sign (for example: xlink:href, svg:x). The elements of main model have only the attribute name. The figure 4 shows a screenshot of AW. The screenshot shows a graphic vision, a textual vision, a modeled workflow and a palette of workflow symbols.

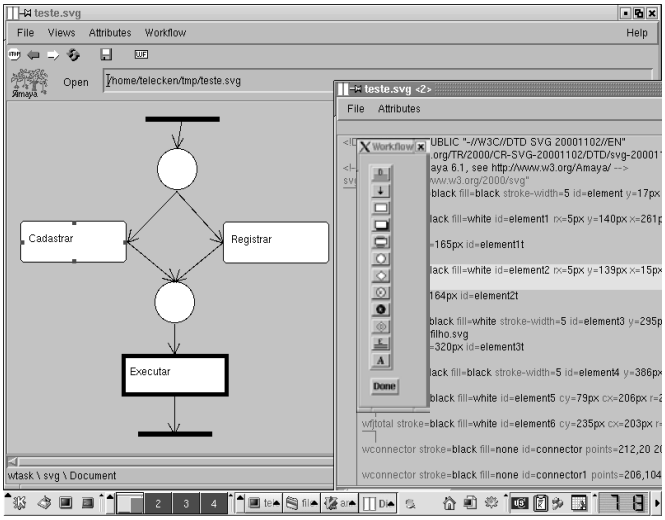


Fig. 4. The AW Screenshot

5.4 The *Many-One* Architecture of AW

The AW process definition can be divided into three distinct domains: workflow domain, graphic domain and link domain. These domains have many mutual relationships and, during the process definition edition, much synchronization is needed. The workflow, graphic and link domains are represented respectively by elements and attributes of AW, SVG and Xlink models.

An integration model was developed to integrate the models. In the integration model, the AW is the main model and the others are the secondary models. The secondary model attributes were inserted into main model elements by namespace standards. The AW process definition is formed by only one document. Into this document there are elements of AW model and attributes of AW, SVG and Xlink models. The figure 5 shows the *many-one* architecture of AW.

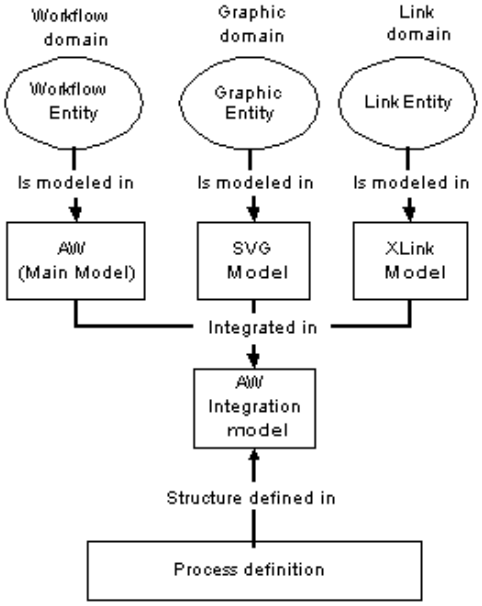


Fig. 5. The AW many-one architecture.

5.5 The Benefits of *Many-One* Architecture in the AW Project

The namespace divides more cleanly the elements of different domains. The elements that represent graphic entities are prefixed by "svg:". The elements that represent links are prefixed by "xlink:". The other elements are elements that represent workflow entities.

XLink and SVG are very diffused official W3C recommendations. In the AW project, documentation and several services available for these models (discussion lists, consultantship, etc) are used. These documentations and services facilitate the model learning, qualify the project and decrease the project costs.

Many applications and tools were developed for SVG and XLink. The AW is compatible and based in these applications and tools. Some features of Amaya system were used and extended in the AW development. The Amaya

system has the following components: a structured document editor that can edit XML document; a SVG viewer and editor; a namespace support and a XLink support. All these components were reused into AW system. Few adaptations were necessary. These reuses decrease the development cost and grow the compatibility of AW with XML, SVG and XLink applications.

The AW development is modular. It is possible to change any model (AW, SVG or XLink) and maintain the others. Also it is possible to add new XML models.

6 *Many-One* Case Study II - Adding Entities in XPDL Representations

In the case study I, an application of *many-one* architecture in one internal representation was described. In this case study, an application of *many-one* architecture in one interchange pattern is described. In this application, SVG attributes are added in XPDL documents.

Following, some XPDL elements are shown:

Package*:* It is a package that contains several processes.
ExternalPackage*:* It references to an external package.
WorkflowProcess*:* It represents a workflow process.
Activity*:* It represents a workflow process activity.
Transition: It represents a transition or a connector between other elements.

6.1 Instructions to Add SVG Attributes in XPDL Elements by the Namespace Standard

In the beginning of internal representation, the namespaces of SVG and AW model need to be defined. The AW is the main model and SVG is the secondary model. An example is shown following:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<Package xmlns="http://www.wfmc.org/2002/XPDL1.0"
xmlns:svg="http://www.w3.org/2000/svg"
version="1.0" Id="0" Name="Sample">
```

In the graphic representation of this case, icons represent the *Activity* elements. There are many types of Activity and for each type there is a correspondent icon. The *Transition* elements are represented by polylines.

Into the XPDL *Activity* elements, the following SVG attributes need to be inserted: *href*, *width*, *height*, *x* and *y* of element *image*. The value of *href* attribute is a URL of a file. The referred file contains an image that represents one type of activity in the graphic vision. The *width, height, x* and *y* elements define respectively the width, height and position of the icon in the graphic vision.

To inform that these attributes belong to *image* element, the *type* attribute with the value " image " needs to be filled. An example of *Activity* element is shown following:

```
<Activity Id="5" Name="Email Confirmation" svg:type=”image” svg:width =”90” svg:height = “40”
svg:x="100" svg:y="100" svg:href=”../activity.jpg”>
    <Implementation>
                <No/>
    </Implementation>
</Activity>
```

Into the XPDL *Transition* elements, the following SVG attributes need to be inserted: *stroke*, *stroke-width* and *points* of element *polyline*.

To inform that these attributes belong to *polyline* element, the *type* attribute with the value " polyline " needs to be filled. An example is shown following:

```
<Transition Id="22" From="1" To="12" svg:type="polyline" svg:stroke="black" svg:points="326,158 320,234
328,221 320,234 313,220" svg:stroke-width="2" >
   <Condition>status == "Valid Data"</Condition>
</Transition>
```

6.2 Adding other entities

An XPDL model can be divided into two distinct domains: the workflow domain and the simulation domain. The elements of simulation domain describe data used by workflow simulation software. The elements of workflow domain describe a workflow and are used mainly by workflow engines.

The simulation elements could be removed from XPDL model. A new model that defines only simulation elements could be created. And the new model elements could be inserted into XPDL documents by namespace standard.

In several XPDL points there are references from URLs. These references are links. The links could be removed from XPDL model and XLink attributes could be inserted into XPDL documents by namespace standard.

The PSL [14] proposes the use of RDF attributes to describe resources used in activities. These attributes could be inserted into XPDL documents by namespace.

An example of Activity code that contain elements and attributes of referred models (XPDL, SVG, XLink, RDF and simulation model) is shown following. The attributes and elements are inserted by namespace standard.

```
<Activity Id="5" Name="Email Confirmation"
      svg:type=”image” svg:width =”90”    svg:height = “40” svg:x="100" svg:y="100"
      xlink:type="simple" xlink:href=”../activity.jpg”
      rdf:resource="email.rdf#confirmation">
    <simulation:SimulationTransformation Instantiation="ONCE">
            <Cost>12<Cost/>
    <simulation:SimulationTransformation/>
    <Implementation>
            <No/>
    </Implementation>
</Activity>
```

The figure 6 shows the architecture of this case.

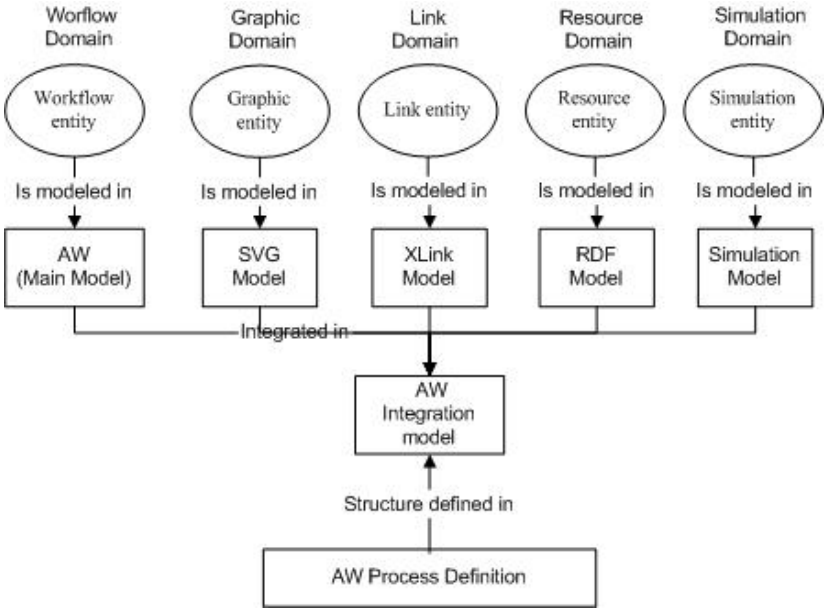


Fig. 6. Case II architecture

The objective of this example was to show the modularity, flexibility and possibility of *many-one* architectures. It is possible to make many other model combinations. Defining the best or more necessary model combination is not approached in this paper but it is an important future work.

7 Comparing the Architectures

The main advantage of the *one-one* architecture is that it is simpler to use because it implements and develops only one model and only one document. However, the current process definitions are very complex documents. There are great demands for inclusion of new elements in the process definition models. For each element that is inserted in the process definition, the model complexity grows. The complexity is propagated for all workflow components directly or indirectly involved with the process definition models (development, implementation, learning and use of workflow technology). And the cost of workflow technology grows too.

In some cases, separating the elements into many models can decrease the workflow technology costs. This is recommended mainly when the elements of model can be divided into distinct domains. The separation is possible in the *many-many* and in the *many-one* architecture. The main advantages of these architectures are:

- The developing of models in an independent way. With an independent development it is possible to divide a big problem into several smaller ones. The big problem is the developing of one model that has elements of several domains. The smaller problems are many models that can be developed in a more independent way and that can be separated into different domains. This independent development also includes all that is developed around the models (application, tools, patterns, learning, training, etc).
- Existent models can be used and reused in a more modular way. The structure of process definition can be formed by elements of different domains, and for each domain the developers can choose to use an existent model or to create a new one. If there are problems with one model, only this model needs to be changed.

The advantages of *many-many* and *many-one* architecture are similar. But the disadvantages are different. The disadvantage of *many-many* architecture is the need to develop resources for document synchronization. The disadvantage of *many-one* architecture is the need to develop an integration model. The table 4 shows the disadvantages and advantages of related architectures.

Table 4 – Architecture's advantages and disadvantages

Architecture	Advantages	Disadvantages
One-One	To use one document and one model (it is the simplest architecture).	To develop in one monolithic way.
Many-Many	To develop models and all technology involved with the models in one independent way. To reuse existent models and technologies in a more modular way.	To develop resources for document synchronization.
Many-One	The same as *many-many*	To develop an integration model.

For choosing architecture, it is important to know two characteristics of the process definition: the domain areas involved and the necessary synchronization among elements from different domains.

The domain areas and the distinction of domain areas are the characteristics that define if it is more appropriate to use an architecture with one model or with many models.

The necessary synchronization among elements of different domains is the characteristic that defines if it is more appropriate to use an architecture with one document or with many documents.

The following recommendations were defined by a comparison among these process definition characteristics and the presented process modeling architecture characteristics:

- If the **distinction of domain areas involved in one process definition decreases**, the use of *one-one* architecture is more appropriated. It is coherent to maintain entities of the same domain in a same model.
- If the **distinction of domain areas involved in one process definition grows**, the use of *many-many* or *many-one* architecture is more appropriated. It is coherent to maintain entities of different domains in different models.
- If the **need of synchronization among elements of different domains grows**, the use of *one-one* or *many-one* architecture is more appropriated. The costs of synchronization are greater when there are elements among different documents.
- If the **need of synchronization among elements of different domains decreases**, the use of *many-many* architecture is more appropriated. The costs of synchronization are greater when there are elements among different documents.

The table 5 shows the application of these recommendations. In the first column the architectures are shown. In the second column the characteristics of process definition appropriated for the associated architecture are shown.

Table 5 – Architecture's recommendations

Architecture	Characteristics of process definition appropriated
One-One	The entity domains are from the same domain
Many-Many	The entity domains are from different domains **and few synchronizations** are necessary
Many-One	The entity domains are from different domains **and many synchronizations** are necessary

8 Applying the Recommendations on Complex Process Definition Models

Many current complex process definitions contain the conditions for *many-one* architecture recommendation. These process definitions contain entity domains from different domains and many synchronizations are necessary in the process definition edition. This is the situation of the case studies shown in this paper. The process definition of the first case study can contain elements or attributes of workflow, graphic and link domain. The process definition of the second case study can contain elements or attributes of workflow, graphic, link, resources and simulation domain. In both case studies, many synchronizations are needed during the process definition edition.

For such situations, the *many-many* is the most onerous architecture. In the process definition edition, the synchronization and maintenance costs are very great for so much relationship among different documents. For using *many-many* architecture it is needed to have no synchronization or few synchronization.

Using the *one-one* architecture provides a more monolithic development. This type of development provides a solution more exact and specific for each application. However the reuse is low. For each application one exact, specific and monolithic solution is needed. It is more difficult to reuse just a part of model or just a part of solution.

If the complexity and the quantity of entities grow very much, a monolithic solution can be very onerous or unviable. In this case, modular solutions such as *many-one* architecture can be more efficient. Modular solutions can divide a great problem into several little ones.

The *many-one* architecture provides a more modular and reusable workflow technology development. For example, in the second case study shown in this paper, groups of developers and researches could work exclusively in the XPDL model. Other groups could work exclusively in each one of other technologies (SVG , XLink, RDF and simulation).

The technology developed by groups dedicated to the SVG, Xlink, RDF and simultation can be reused by: (i) other interchange patterns, such as BPML and BPEL4WS;(ii) other internal representations, such as Biztalk Orchestration Designer internal representation; and (iii) any other application inside or outside of workflow area, such as SVG viewers and Extensible Hypertext Markup Language (XHTML) links.

Integration groups also are important. These groups make the integration of different modules and technologies. In the first case study, a group that make the integration of AW, SVG and XLink is needed. In other projects, other groups can integrate BPEL4WS with XLink, PSL with RDF (this is the case of PSL [14]), XPDL with SVG , XLink, RDF and simulation (this is the situation of second case study), etc.

9 Conclusion

Following, some *many-one* recommendations are presented. These recommendations are a summarization of the main contributions of this paper:

1. It is recommendable to use the *many-one* architecture in a complex process definition application that contains entity domains from different domains and when much synchronization among elements from different domains is necessary. Many current process definitions have these characteristics. Many complex environments need process definition with such characteristics.
2. When a more modular and independent development is needed, it is recommendable to use or to consider the *many-one* architecture.
3. For optimizing the *many-one* architecture benefits it is recommendable to reuse current XML models and reuse all technology developed around these models (application, tools, APIs, services, documentation, resources, researches, involved community, implementation, technologies, etc).
4. As the use of *many-one* architecture (mainly using these recommendations) grows, the *many-one* architecture benefits grow too. In an ideal scenario there are many models for different domains. There are many integration developers and researches that can group and regroup the available models (and the involved technology) in according with the specific application needs.

The ideal proposed scenario is not a distant scenario. In the current days, there are many XML models available. The model integration technology is available too. Some applications using namespace architecture already were implemented.

But it is necessary to organize and to optimize this scenario. This can be made by more researches about model integration technologies, the use of XML models in process modeling phase and process modeling architectures.

Finally, it is expected that this paper: (i) have presented the main ideas, fundaments and recommendations about *many-one* architecture; and (ii) make developers, researches and organizations, such as WfMC and BPMI, aware about the importance of optional process modeling architectures. In complex environments, the three solutions (*one-one, many-one* and *many-many*) need to be considered.

Acknowledgements

The authors want to tank the support of CNPq, UFRGS, INRIA and Banco Central do Brasil.

References

[1] van der Aalst, W.M.P. Don't go with the flow: Web services composition standards exposed, *IEEE Intelligent Systems*, 18(1):72-76, 2003.
[2] BPMI, Business Process Management Initiative Home Page. http://www.bpmi.org
[3] Bray, T.; Hollander, D. and Layman, A. Namespace in XML, W3C Recommendations, 1999. http://www.w3.org/TR/REC-xml-names.
[4] Bray, T.; Paoli, J.; Sperberg-McQueem, C.M. and Maler, E. eXtensible Markup Language (XML) 1.0 (Second Edition), 2000. http://www.w3.org/TR/REC-xml.
[5] Casati, F.; Grefen, P.; Pernici, B.; Pozzi, G. and Sanchez, G. WIDE Workflow Model and Architecture, Technical Report 96-19, Centre for Telematics and Information Technology (CTIT), University of Twente, Netherlands, 1996.
[6] DeRose, S.; Maler, E. and Orchard, D. XML Linking Language (XLink) Version 1.0. W3C Recommendation, 2001. http://www.w3.org/TR/xlink/
[7] Dodds, D.; Watt, A.; Birbeck, M.; Cousins, J.; Moore, D.R.; Worden, R. ; Nic, M.; Ayers, D.; Ahmed, K.; Wrightson, A. and Lubell, J. Professional XML Meta Data, Wrox press, Birmingham, AL, 2001.
[8] FORO, Models Designer Manual, 2001. http://www.foro-wf.com/docs/english/ModelDesigner2.1.3.pdf
[9] IBM. IBM MQ series Workflow Programming Guide Version 3.3, IBM Corporation, Armonk, USA, 2001.
[10] ILOG Inc, Ilog components for business process management solutions, 2001. http://www.ilog.com/products/jviews/workflow/workflow_wp.pdf
[11] Leymann, F. Web Services Flow Language (WSFL), Technical report, IBM, 2001.
[12] Leymann, F. and Roller, D. A quick overview of BPEL4WS, IBM DeveloperWorks, August 2002.
[13] Pinheiro, M.K.; Telecken, T.L.; Lima, J.V.; Zeve, C.M.D. and Edelweis, N. A Cooperative Environment for E-Learning Authoring. Document Numérique, França, v.5, n. 3-4, p. 89-114, 2002.
[14] Schlenoff, C.; Gruninger, M.; Tissot, F.; Valois, J.; Lubell, J. and Lee, J. The Process Specification Language (PSL): Overview and version 1.0 specification, NISTIR 6459, National Institute of Standards and Technology, Gaithersburg, MD, 2000.
[15] Sheth, A.P.; Georgakopoulos, D.; Joosten, S.; Rusinkiewicz, M.; Scacchi, W.; Wileden, J.C. and Wolf, A.L. Report from the NSF Workshop on Workflow and Process Automation in Information Systems, SIGMOD Record 25(4) (1996) 55-67.
[16] Telecken, T.L.; Lima, J.V.; Zeve, C.M.D.; Maciel, C. and Borges, T. Modeling of Courses through Workflow using the standard SVG/XML. In Proceedings of EDMEDIA'2002 World Conference on Educational Multimidia, Hipermidia & Telecomunications, Denver, USA, 2000, 24-29
[17] Thatte, S. XLANG. Web Services for Business Process Design, Technical report, Microsoft Corporation, 2001.
[18] Workflow Management Coalition Home Page. http://www.wfmc.org
[19] Workflow Management Coalition. Interface 1 - Process Definition Interchange. Technical report WFMC-TC-1025, 2002.
[20] Workflow Management Coalition. Terminology and Glossary, Technical report, WFMCTC-1011, Brussels, 1996.
[21] World Wide Web Consortium. Resource Description Framework (RDF), W3C Recommendation, 1999. http://www.w3.org/TR/REC-rdf-syntax
[22] World Wide Web Consortium. Scalable Vector Graphics (SVG) 1.0 Specification. W3C recommendation, 2001. http://www.w3.org/TR/SVG/
[23] World Wide Web Consortium. Web Service Choreography Interface 1.0, 2002. http://www.w3.org/TR/wsci/
[24] World Wide Web Consortium. Welcome to Amaya. http://www.w3.org/Amaya/

Why Programmer-specified Aliasing is a Bad Idea

Markus Mock
University of Pittsburgh, Department of Computer Science
Pittsburgh, PA 15260, USA
mock@cs.pitt.edu

Abstract

The ISO C standard C99 has added a special keyword, named `restrict` to allow the programmer to specify non-aliasing as an aid to the compiler's optimizer and to thereby possibly improve performance. However, it is the programmer's responsibility to ensure that the annotations are correct. Therefore, in practice, `restrict` will be useful only when the programmer's effort is rewarded with noticeable performance improvement. To assess the performance potential of the `restrict` annotation, we automatically generated best-case `restrict` annotations for SPEC CPU2000 benchmarks by using pointer profiling. However, even though we used the best possible `restrict` annotations, we found an average program speedup of less than 1% on average when using two state-of-the art optimizing compilers that implement the `restrict` pragma. Since the typical performance benefits do not warrant significant user effort and potential errors, we conclude that having the programmer specify non-aliasing is a bad idea.

Keywords: C, C99 standard, restrict, compilers, program optimization, aliasing, dynamic points-to analysis

1 Introduction

In most programming languages it is possible to access a memory location in multiple ways using different names. For instance, in programming languages that support call-by-reference parameters, the name of the formal and actual argument refer to the same location and are therefore aliased. Aliases do also frequently arise due to pointers to named variables: if `p` points to variable `x`, both `x` and the pointer dereference `*p` access the same memory location.

To ensure that correct code is generated in the presence of aliases, compilers have to perform an *alias analysis* to determine the aliases in the program and disambiguate memory accesses. Consider the example in Figure 1, which shows the fragment of a C program. Procedure `vmul` takes two arrays of floating point numbers, `a` and `b` and their size `n` as inputs and computes the square of each element of `a` and stores it element by element in array `b`. Without further knowledge and without special hardware support, the compiler must assume that `a` and `b` may refer to the same array or overlapping parts of an array so that the loop cannot be parallelized or software-pipelined because it has to be ensured that an update of `b[i]` is performed before the next value `a[i+1]` is loaded. This constraint also prevents the compiler from generating loads for multiple array elements of `a` at the same time since the subsequent store to `b[i]` may modify elements of array `a`.[1]

To compute the aliases in a program, many algorithms have been developed. Flow- and context-sensitive algorithms potentially produce the most precise results, but they generally do not scale well, which limits their applicability to relatively small programs (50,000 lines of code at most). In addition, recent work [5] suggests that for many C programs flow- and context-sensitivity produce only insignificant improvements over Das's fast One-Level Flow algorithm [4], which has been shown to scale up to over a million lines of C code.

Because flow- and context-sensitive pointer analyses are usually too expensive to be used in production compilers, often only very simple pointer analysis is performed in practice. For instance, the analysis may simply assume that a pointer dereference may define any variable that has its address taken in the program. Unfortunately, the conservative results ensuing from such simple algorithms may prevent the compiler from performing aggressive code optimization.

Even though more expensive algorithms potentially produce better results, compared to actual run-time pointer behavior their results are still very conservative. Mock et al. [19] showed that while the points-to sets of SPEC CPU2000 benchmarks computed by several well-known scalable points-to analyses typically contained hundreds or even thousand of potential pointer targets, at run-time pointers typically point to only a few, in fact in most cases only one, logical location.[2] This means that even compilers that use more sophisticated pointer analysis algorithms may in

[1] When the processor provides mechanisms for data speculation, the compiler may decide to speculatively load the value into a register and rely on the hardware to detect whether the value has to be reloaded. For example, on the Intel Itanium processor the compiler can use an advanced load instruction in conjunction with a check instruction to perform this kind of data speculation (cf. Section 4.2 for more details).

[2] A logical location is either a program variable or a heap allocation site. There may be multiple instantiations of a single logical variable in the case of local variables and multiple distinct objects allocated at the same memory allocation (heap) site.

```
void vmul(int n, double* a, double* b) {
   int i;
   for (i=0; i<n; i++)
      b[i] = a[i] * a[i];
}
```

Figure 1: A simple vector multiplication routine. If it is known that arrays `a` and `b` are not aliased, the compiler can software-pipeline or parallelize the loop.

```
void vmul(int n, double* restrict a, double* restrict b) {
   int i;
   for (i=0; i<n; i++)
      b[i] = a[i] * a[i];
}
```

Figure 2: The vector multiplication routine whose arguments have been annotated with the `restrict` pragma to indicate to the compiler that the arguments are not aliased, thereby allowing the compiler to perform more aggressive optimization, such as software pipelining. If the annotation is incorrect, the resulting code may be incorrect.

practice be prevented from performing some optimizations even when those would be sound in reality.

Pointer analysis imprecision is a particularly severe problem for the C programming language, where widespread pointer use, a weak type system, and pointer arithmetic often lead to very imprecise analysis results. This has been long recognized by C users and compiler writers alike and led to a community effort to add a language pragma to the C language standard that could be used by C programmers to communicate to the compiler that certain expressions in a program are not aliased. The latest C99 standard (formally defined as ISO/IEC standard 9899:1999 [13]) introduces a `restrict` keyword, which can be used to qualify formal pointer-type arguments of procedures.[3]

The definition of the `restrict` keyword specifies that "an object that is accessed through a restrict qualified pointer requires that all accesses to that object use, directly or indirectly, the value of that particular restrict qualified pointer. Any access to the object through any other means may result in undefined behavior. The intended use of the restrict qualifier is to allow the compiler to make assumptions that promote optimizations." (quoted from the README file distributed with the Sun C compiler, which implements the C99 standard, one of the compilers used in this study).

Figure 2 shows an example of how `restrict` is supposed to be used. The two pointer-typed arguments `a` and `b` are qualified with the `restrict` keyword to tell the compiler that the array pointed to by `a` is only accessed via pointer `a` and any pointers derived from it (e.g., accesses such as `a[i]` which use an address obtained by adding an offset to `a`). Similarly, the array pointed to by `b` is only accessed via pointer `b` and pointers derived from it. In particular, these assertions state that `a` and `b` may not point to overlapping or identical arrays. Consequently, based on these assertions, the compiler can perform (software) parallelization and other transformations whose correctness depend on `a` and `b` not being aliased. Compared to the routine shown in Figure 1, the restrict-qualified routine in Figure 2 executes 24% faster on a Sparc workstation and even 2.9 times faster on an Itanium-based workstation with two processors.[4]

Since the `restrict` keyword is an assertion specifying that the qualified function arguments are not aliased at run time, it has to be used with utmost care because the compiler may perform some optimizations that are not sound when the arguments are in fact aliased. While a compiler user who is familiar with the program he/she is compiling may be able to use the `restrict` keyword correctly, it will still be a tedious task and, without any support by an automatic tool, likely be error prone. In practice, therefore, users will only accept this tedium if they can expect a noticeable performance gain. While previous work [3] found that in some cases memory disambiguation can result in significant speedups for kernel array codes, it is not clear what performance gains are achievable with the `restrict` pragma on general C programs of realistic size.

In this paper we show that even optimistic annotations with `restrict` pragmas (i.e., annotations that may only be sound for some but not all inputs) lead only to minor performance improvements for C programs of significant size and variety. We were able to verify that this result appears to hold across compilers and architectures by using two distinct commercial optimizing compilers on two different processors (Sparc V9 and Itanium 2) and obtaining very similar results. Therefore, we conclude that is generally a bad idea to put the specification of non-aliasing into the programmer's hands.

However, while the typical performance improvements are too small to warrant extra programmer effort, we would like to realize those improvements if this could be done with no or insignificant programmer effort. Fortunately, this is

[3]The C standard also defines the use of `restrict` on structure fields to specify that `restrict`-qualified fields are not aliased to other program expressions.

[4]The Sparc speedup was measured on a Sun Blade workstation with version 7 of the Sun Studio One C compiler and highest (-xO5) optimization level; the speedup for the Itanium processor on a HP ZX6000 machine with 2 Itanium 2 processors, and Intel's ecc compiler with highest (-O3) optimization. In both cases, a vector size of 100 was used and the loop was executed 1,000,000 times.

feasible: with simple pointer profiling, `restrict` annotations can be derived automatically. We present an approach that ensures that such annotations are sound by generating a run-time check that redirects execution to conservatively optimized code (assuming aliasing) when the run-time check fails.

In more detail, this paper makes the following contributions:

1. By observing alias relationships at run-time and using them to place `restrict` pragmas, we obtain an upper bound on the potential performance improvement arising from `restrict` pragmas in two real highly optimizing C compilers;

2. we show that for a large class of diverse, compute-intensive applications (SPEC CPU2000 benchmarks), only minor improvements can be achieved – on average less than 1% and no more than 8% in our experiments;

3. we demonstrate how `restrict` pragmas generated by run-time observation of alias relationships can be made sound across all inputs by outlining how to generate guard code that selects a conservatively optimized version of a function when aliases occur at run-time;

4. and finally, we show that placing `restrict` pragmas based on static alias analysis alone is in general much less effective than generating guarded `restrict` pragmas using run-time information.

The rest of this paper is organized as follows: Section 2 gives some background on pointer analysis and describes how we obtained the dynamic alias information used for our optimization experiments. Section 3 describes our experimental setup and the workload used in this study. Section 4 presents the results of our experiments and analyzes the reasons for the observed speedups. In Section 5 we propose an automatic approach to generate sound uses of `restrict` pragmas, and in Section 6 we discuss related work. Finally, Section 7 presents conclusions and directions for future research.

2 Alias Analysis

In the C programming language aliases can arise in two ways. First, different fields of a union data structure are aliased to each other. Second, variables that have their address taken can be accessed both via their name and a pointer dereference. Aliases arising via union data structures are easily identified, potential aliases via pointers, on the other hand, require a pointer analysis.[5] This analysis is often performed as a *points-to analysis*, which computes for each pointer `p` and pointer dereference expression `exp` the set of logical locations that may be accessed via `p` or `exp`. Some pointer analyses compute the set of possible aliases due to pointers directly (e.g.,Landi et al.'s algorithm [16]).

2.1 Points-to Analysis

Aliases can be computed easily from points-to sets: given variable x and pointer-valued expressions $exp1$ and $exp2$, x is aliased to $*(exp1) \iff x \in pts(exp1)$, where $pts(exp)$ denotes the points-to set of expression exp, and $*(exp1)$ and $*(exp2)$ are aliased $\iff pts(exp1) \cap pts(exp2) \neq \emptyset$.[6]

Traditional *static* points-to analyses compute an approximation of the set of objects to which a pointer may point. They are conservative in the sense that their results must be correct for any input and execution path of the program. In addition, for the C programming language they have to make various conservative assumptions when analyzing a program, for instance, because of C's weak type system.

An alternative way of gathering points-to data is to perform a *dynamic* points-to analysis. A dynamic points-to analysis records the targets of program pointers during actual program execution, by instrumenting the program source with calls to an appropriate data-capturing routine. Since dynamic points-to sets only capture the targets of pointers during a particular program execution, they are in general unsound (i.e., optimistic). Mock et al. [19] showed that the typically observed dynamic points-to sets are 10–100 times smaller than the points-to sets computed by Das's highly-scalable One-Level Flow algorithm [4], which generally produces results as precise as Andersen's well-known algorithm [2]. Andersen's algorithm, in turn, has been shown [17] to be of comparable precision as some other well-known pointer analysis algorithms [17, 20]. Mock et al. [19] additionally showed that the majority of program variables in the SPEC CPU2000 benchmarks point to only a single logical location during execution with the SPEC-provided test inputs. Although more expensive flow-sensitive algorithms [22] can obtain better average points-to sets than the scalable analyses used in [19], they still do not in general yield points-to sets as small as the dynamic sets.

[5] Sometimes the terms *pointer analysis*, *points-to analysis*, and *alias analysis* are used interchangeably in the literature. In this paper, we use the term alias analysis for an analysis that determines for a memory-access expression the set of other expressions (variable names or pointer dereferences) that may refer to the same memory location.

[6] This rule does not take aliases via union data structures into account. For the rest of the paper we do not distinguish fields of structures or unions, i.e., a reference of a structure or union field is treated as if the whole structure or union were referenced. Since distinguishing structure and union fields in a pointer analysis can potentially make the algorithm much less efficient, many pointer analyses do not distinguish them, e.g., [4, 23].

	Source Lines	Reachable Functions	Description
art	1,270	22	image recognition, neural networks
equake	1,513	27	seismic wave propagation simulator
mcf	1,909	24	combinatorial optimization
bzip2	4,639	63	compression
gzip	7,757	62	compression
parser	10,924	297	word processing
ammp	13,263	161	molecular dynamics
vpr	16,973	255	circuit placement and routing
twolf	19,748	167	placement and global routing
vortex	52,633	643	object-oriented database
mesa	49,701	770	graphics package
gap	59,482	826	group theory interpreter

Table 1: Sizes and descriptions of the programs used in the experiments.

Since all logical locations contained in a dynamic points-to set must also be contained in its static counterpart, using dynamic points-to sets to compute aliases in a program enables us to obtain a lower bound on the aliases present in a program. Using these optimistic aliases in the compiler's optimization phase therefore provides an upper bound on the improvement we can hope to achieve by using `restrict` pragmas in a program. Obviously, the `restrict` pragmas obtained by this method are optimistic, i.e., only guaranteed to be sound for the program input that was used to compute the alias relationships. For a bound on the potential impact on optimization, however, this is immaterial. When used for optimization in practice, however, we must guarantee that the `restrict` pragmas are sound for all inputs; Section 5 outlines how to generate run-time guards to make `restrict` pragmas safe and enable aggressive optimizations at the same time.

2.2 Dynamic Alias Analysis

To generate the dynamic alias information used to place `restrict` pragmas, we used a slightly modified version of the instrumentation tool *Tumi* [18, 19], which works in three steps. First, a static points-to analysis is run on the application source code. For each pointer and dereference point, it computes a conservative approximation of the set of logical locations a pointer may point to.

Second, the application is instrumented, inserting a call to a run-time routine (at the entry of each procedure) that, for each procedure argument `arg`, matches the address contained in `arg` with the run-time addresses of potential pointer targets for `arg` (identified by the static points-to analysis of the first step). [7] At run-time, when the run-time library routine identifies the same logical location for two distinct arguments, the arguments are marked as aliased.[8]

As the final step, the instrumented application is compiled, and executed on some input. Upon termination, the instrumentation code saves a record of the aliasing relationships that occurred during execution. In this process the address matching step is essential: since distinct run-time addresses may refer to the same logical location, simply recording the pointer addresses is not sufficient to construct the aliasing relationships at run time. More details about the instrumentation algorithm can be found in [18, 19].

To obtain the dynamic aliasing relationships for the applications in this study, we used the SPEC-provided reference inputs to ensure that the resulting `restrict` pragmas were sound for the timing runs, which use the reference inputs. Das's One-Level Flow algorithm [4] was used as the static points-to analysis of the first step.

3 Experimental Setup and Workload

To determine the performance impact of `restrict` pragmas on realistic programs, we chose to use applications from the SPEC CPU2000 benchmark suite since SPEC benchmarks are of considerable size, cover a wide range of tasks (e.g., simulations, group theoretic computations, graphics, databases, word processing), are actually used in practice, and are generally used to evaluate CPU performance, i.e, they are generally considered to be good indicators of CPU and compiler performance. Table 1 shows the programs used with their sizes (in lines of C code) and the number of statically reachable functions, for which we generated `restrict` annotations.

We performed our experiments on two different platforms. The first platform was a Sun Blade-100 workstation with a 500MHz UltraSPARC-IIe processor, 256 MByte of RAM, and running Sun OS 5.8 (Solaris). For this platform

[7] Since arguments that have empty (static) points-to sets are guaranteed not contain pointers at run-time, only arguments with non-empty static points-to sets are instrumented.

[8] In addition, for the `restrict` annotation to be correct, other memory references inside the procedure must also not be aliased to the procedure argument. In our experiments, we verified that this was the case by manual inspection of all dynamic points-to sets within a procedure.

	Un-aliased at run time	Aliased statically
equake	26	16
mcf	3	2
parser	22	18
ammp	12	2
vpr	50	35
twolf	11	7
vortex	397	397
mesa	18	18
gap	49	48

Table 2: Column two shows the number of functions for which none of the arguments were found to be aliased at run-time; column three the number of those that were reported to be aliased by the static pointer analysis. For instance, for `vpr`, there were 50 functions with no aliases in the arguments at run time; out of those 50, 35 were reported to have aliased arguments by the static points-to analysis. `art, bzip2` and `gzip` are not shown in the table since the static alias analysis was able to determine that no aliases were present in these applications for the procedure arguments.

we used the Sun Studio One C compiler, version 7, and compiled the benchmarks with the highest optimization option -xO5, using cross-file optimization (-xcrossfile) and the -fast option, which turns on a couple of optimizations designed to improve execution speed (e.g., data alignment along double word boundaries to improve memory access times).

The second platform was a Hewlett Packard ZX6000 workstation with two 900 MHz Itanium 2 processors, 1 GB of RAM, and running Redhat Linux 7.2. For this platform we used the ecc compiler from Intel, version 7.0 at highest optimization level (-O3), with interprocedural optimization across files (-ipo), and the -restrict option to enable the use of the `restrict` qualifier.

For both platforms, all timings were obtained using the `runspec` script provided with the SPEC CPU2000 benchmark suite. We ran the benchmark several times on otherwise unloaded machines and took the best (shortest) execution time of all runs. We used the reference input data sets, which represent the largest input data sets available in the SPEC benchmark suite. We did not use feedback directed optimizations in our experiments to avoid any interaction of profiling with the `restrict` optimization.

4 Results

4.1 Aliasing Results

Table 2 shows for each benchmark the number of reachable functions with at least one pointer-type argument (on which the `restrict` pragma might be used) and for which none of the arguments was found to be aliased at run-time. Column three, for comparison, lists the number of those functions for which the static pointer analysis (Das' algorithm [4]) reported that some arguments might be aliased. For instance, for `gap` 49 functions with at least one pointer-type argument did not have any aliases at run-time in their arguments. For 48 of those 49 functions, however, the static points-to analysis reported that the arguments might be aliased. With the exception of `mcf` and `ammp`, generally the alias information produced by the static pointer analysis was not a very accurate representation of the run-time alias relationships. This shows that there is an opportunity for exploiting run-time alias information in practice, but how effectively can it be realized with `restrict` pragmas?

4.2 Best Case Speedups with restrict

To answer this question, we generated the `restrict` pragmas, compiled and executed the programs as described in Section 3. Figure 3 shows the speedup for each application when compiled with `restrict` pragmas versus the execution time of the same program that was compiled without any `restrict` pragmas. In both cases, all other optimizations options were identical (`-xO5 -fast -xcrossfile` for the Sparc platform and -O3 -ipo -restrict for the Itanium platform).[9] Since no aliases where found by the static pointer analysis for `art, bzip2` and `gzip`, they have been omitted from the graphs.

On average (geometric mean) across all benchmarks, we see only a modest improvement of just a little under 1% on the Sparc platform; on Itanium, the average was close to one. The best speedup improvement was achieved for application `ammp` on the Sparc platform, which improved by 7.5%.

While `ammp` was the benchmark with the largest speedup on the Sparc platform (in fact, the largest speedup in general), its performance was identical to the unannotated version on the Itanium platform. We even found that the

[9] In the Sun Studio One C compiler recognition of the `restrict` pragma is on by default, for the Intel ecc compiler it has to be turned on explicitly with the -restrict switch.

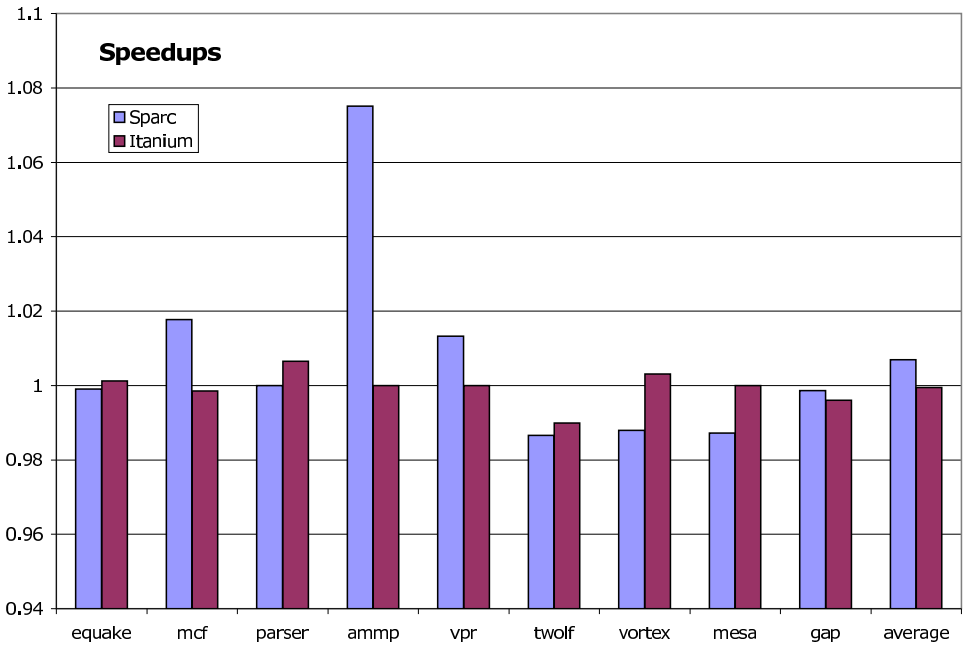


Figure 3: Upper bound speedups of the applications obtained by placing `restrict` on all arguments that were found to be unaliased when run on the reference input.

ecc compiler produced identical binaries for the benchmark with and without `restrict` pragmas. Since in other benchmarks (and as mentioned in Section 1) ecc actually is able to take advantage of the pragma we looked more closely at the binary that was generated.

The compiler performs aggressive data speculation exploiting the advanced load instruction on the Itanium (for instance `ldfd.a`) which loads a value into a register and stores the memory address from which the data was loaded in the *Advanced Load Address Table* (ALAT) hardware structure. Before the use of a value loaded with an advanced load instruction, the compiler inserts a `chk.a` instruction. This instruction will branch to fix-up code to reload the value if its load address is not in the ALAT and recompute any speculatively computed values based on the value loaded with the advanced load. Since the processor automatically removes an entry from the ALAT when a store to an address in the ALAT occurs, this ensures correctness in case of an intervening store. Using this feature, the compiler apparently found no additional optimization opportunities with the `restrict` annotations beyond the normal data speculation that is already performed for possibly aliased loads.

We ascertained this explication using the Itanium processor's performance counters. They showed that during the execution approximately 35 million check instructions were execution of which less than 0.04% failed the check, i.e., required a reload from memory. That is, the compiler is able to overcome poor static alias information by using the Itanium's hardware support for data speculation, breaking the pointer-induced data dependences, and so aggressively optimizing the application.

For the Sparc platform, `mcf` and `vpr` were the only other benchmarks for which the `restrict` pragma resulted in speedups. For several of the benchmarks, the code with `restrict` actually ran somewhat slower. We believe this to be due to degraded instruction cache performance since code compiled with `restrict` tends to be considerably larger than the original code because the compiler performs loop parallelization, aggressive unrolling etc. which can improve run-time but result in larger instruction cache footprints.

For the Itanium platform, `equake` and `parser` also showed minimal speedups, however, below 1%. For `vortex`, the `restrict` annotation led to a performance degradation of about 1% on the Sparc platform, and minimal improvement on the Itanium platform. `gap`'s performance was virtually unchanged on both platforms with an insignificant degradation in both cases.

5 Sound, Aggressive restrict Annotations

The results in Section 4 indicate that there is generally too little performance gain from `restrict` annotations to justify the additional programmer effort and potential for errors by incorrectly placed annotations. However, some applications may benefit noticeably (e.g., `ammp` on the Sparc platform in our experiments), and for others even minor improvements may be welcome if they can be obtained without additional programmer effort.

However, to realize those improvements where possible without user intervention, `restrict` annotations placed by an automatic tool have to be sound. The data shown in Table 2 demonstrates that a tool relying on static (pointer) analysis alone will generally not be able to place `restrict` annotations aggressively and would usually not be able to effectively exploit non-aliasing. Yet, annotations generated exclusively based on dynamic alias information, which is guaranteed to detect non-aliasing when present, is unsound since it cannot guarantee the absence of aliasing across all inputs. Therefore, we propose the following approach that combines dynamic and static analysis with run-time checks to generate aggressive yet sound alias annotations.[10]

To ensure the soundness of a routine some of whose arguments have been qualified with `restrict`, guard code is added, which checks at run-time whether the arguments are in fact alias-free. If an alias is detected at run-time, the guard condition dispatches to a conservatively optimized version of the routine, otherwise the aggressively optimized routine (optimized assuming no aliasing) is executed. Figure 4 shows an example.[11] The original `vmul` routine has been replaced by a routine that checks for an alias for `a` or `b`. If the check returns `true` (no alias is present), then the routine that was optimized with the `restrict` annotation is called (renamed to `vmul-r`); otherwise `vmul-s` is called, the original `vmul` routine without any `restrict` pragma (i.e., optimized assuming potential aliasing). Effectively, we have specialized routine `vmul` for the presence or absence of aliases. To avoid additional call overhead, the `vmul` routine, which only serves as a trampoline to select the correct specialized code version, will be inlined into its caller (indicated by the `inline` keyword in the example). If the transformation is performed at the intermediate code level, this inlining could be performed at compile time if all code is available then, or at link time when the code for all the callers of `vmul` becomes available.

Since the run-time check `check_vmul_alias` adds an additional run-time cost to the original routine, it is important that its cost be recouped by sufficiently often executing the more aggressively optimized version. To determine whether the transformation is worthwhile, the compiler should assess the likely ensuing benefit, for instance by using an approach similar to the inline trials used by Dean et al. [7]. At the same time, it is important to make the run-time alias check as cheap as possible.

Our instrumentation tool Tumi provides library routines that can be linked with an application to automatically detect aliases at run-time. The same steps used to identify the logical location pointed to by a pointer can be used to check whether aliasing occurs: if address matching for the possibly aliased locations returns the same logical location, aliasing is present and the conservatively optimized code version can be dispatched, otherwise the aggressively optimized version. Unfortunately, currently this requires that the application be run with some instrumentation code in place, which degrades performance considerably.[12] Looking at possible hardware mechanisms to make the instrumentation more efficient, is an interesting area for future research.

While using Tumi's general alias detection mechanism is typically too expensive, in many cases much cheaper tests can be generated as guard code. If, for instance, the pointer analysis can determine statically that aliases with `a` and `b` will either not occur or that that the two arguments will point to the same base address, `check_vmul_alias(a,b)` can be implemented as a simple pointer comparison of `a` and `b`. How to generate those cheaper tests automatically and with minimal additional run-time (pointer profiling) overhead, is part of future research.

6 Related Work

Bernstein et al. [3] use a static memory disambiguator for array references to generate run-time conditions that check whether an alias actually occurs at run-time in those cases where the static analysis cannot rule out aliasing. In their approach, tests for these alias conditions are inserted into inner loops and (statically) two code versions are generated for the loop: one optimized assuming no aliasing and one with aliasing. To minimize the run-time cost of selecting the appropriate code version, they perform their transformation only when the dynamic condition is loop-invariant so that it can be hoisted out of the loop. On the downside, limiting their transformation to this subset of all inner loops also diminishes the applicability of their approach. On integer codes they found no improvements, and with the exception of two applications, floating point benchmarks also showed no speedups. The two applications that showed significant speedups (`alvin` 117% and `ear` 37%), are very small compared to the SPEC CPU2000 benchmarks; `alvin` is 200 lines and `ear` 3,000 lines of C code, i.e., really significant speedups were only achieved for kernel-size applications.

Postiff et al. [21] investigated hardware support to enable the promotion of a variable from memory to a registers. In the presence of an alias, this can generally not be done (at least without repeatedly reloading the register which defeats the optimization purpose). They propose a combined compiler-architecture approach, where the compiler

[10] Note that this idea per se is not new, and has been applied in similar fashion by several researchers [3, 6, 11]. While previous approaches typically relied on heuristics to decide when code speculation based on aliasing should be performed, the approach proposed here is based on dynamically gathered alias profiles that represent likely alias relationships at run time.

[11] The example shows the transformation in C code; in practice, the transformation would most likely be performed at the compiler's intermediate representation level.

[12] For all benchmarks in this study, the slowdown incurred by Tumi's instrumentation was over 10% [18]. Therefore, none of the best-case speedups achieved with `restrict` lead to actual speedups when Tumi's current alias detection mechanism to dispatch to the appropriate code version at run time is used.

```
inline void vmul(int n, double*  a,  double* b) {
   if (check_vmul_alias(a,b))
       vmul-s(n, a, b); /* aliased */
   else
       vmul-r(n, a, b); /* not aliased */
}
void vmul-r(int n, double* restrict a, double* restrict b) {
   int i
   for (i=0; i<n; i++)
      b[i] = a[i] * a[i];
}
void vmul-s(int n, double* a, double* b) {
   int i
   for (i=0; i<n; i++)
      b[i] = a[i] * a[i];
}
```

Figure 4: Guard code example. The original routine `vmul` from Figure 1 has been converted into a trampoline routine that is inlined into callers of `vmul`. Its only purpose is to select the correct code version of `vmul` at run time: `vmul-r`, i.e., `vmul` optimized with `restrict` annotations is selected if the run-time non-aliasing check for the `restrict`-annotated arguments succeeds; otherwise, `vmul-s` the safely optimized version of `vmul` is selected. By inlining the `vmul`-trampoline routine no additional function call overhead is added. If the alias check `check_vmul_alias` is simple enough (for checking that arrays `a` and `b` are not aliased a simple pointer comparison may suffice), it can also be inlined, further reducing the run-time checking overhead.

---

loads the address of a promoted variable into a special processor data structure (the store-load address table). When aliased accesses to the variable occur, they are automatically intercepted by the processor and the value is forwarded directly into the register into which the variable was promoted. In their simulation they found a moderate reduction in the number of loads and stores on average, however, it is unclear what the bottom-line performance effect of their combined approach might be in practice. Note however, that the proposed hardware structure is very similar to the ALAT structure on the Intel Itanium processor [12].

Das et al. [5] compare several static pointer analyses to estimate the potential impact on optimization arising from more precise static pointer analysis. They demonstrate the number of aliases reported by fast control-flow insensitive analyses is not significantly different from the number reported by more expensive flow-sensitive algorithms. Their works differs from ours inasmuch as they do not look at the potential for performance improvement arising from compiler pragmas (such as `restrict`), nor do they compare concrete execution times resulting from different alias analysis precision.

Ghiya et al. [10] do exactly this by comparing the run times of SPEC benchmarks when compiled with different memory disambiguation algorithms in their compiler for the IA-64 (Itanium) architecture. Similar to Das et al., they found that more expensive algorithms in most cases did not provide additional benefits. In practice, a more important property for the optimization of their C benchmarks was whether or not the algorithms distinguished structure fields.

Foster and Aiken [1, 9] define a precise semantics for the `restrict` qualifier and then present a type analysis that automatically infers whether `restrict` annotations written by a programmer obey the semantics. Their goal is to detect violations of certain program correctness properties, for instance, that a lock is not acquired again before releasing it. While their approach might benefit from dynamic alias information, in many cases the static alias information seems to be sufficient for detecting programming errors.

Koes et al. [14, 15] present work similar to ours. They propose another programmer annotation similar in spirit to `restrict`, that allows the programmer to specify the non-aliasing of two arbitrary pointers and study its usefulness for performance optimization. They also use profiling to determine when the annotations would be useful. However, they do not present an automatic scheme to ensure soundness and their performance numbers are only simulated and therefore have to be taken with a grain of salt. Their simulation results, however, show the same trend: for most applications there was only minimal improvement, and best-case (simulated) speedups were under 30% in all cases for realistic processor assumptions.

Fernandez and Espasa [8] present an approach for performing alias analysis on executable code at link time. In addition, they mention using run-time profiles to perform speculative optimization, without however, describing an automatic approach to achieve soundness.

Huang et al. [11] perform speculative disambiguation for the LIFE VLIW architecture. They produce specialized code that disambiguates memory at run-time, similar in nature to the approach outlined in Section 5. The difference lies in the granularity of the approach: they look at individual memory dependences (e.g., a RAW (read-after-write) dependence) for individual load and store instructions and build a dependence tree to determine the affected instructions. At run-time, either the speculative code or conservatively optimized code (obeying the potential dependence) is executing using the VLIW's predication mechanism.

Davidson and Jinturkar [6] use a software-based dynamic memory disambiguation approach to enable aggressive

loop unrolling (to increase instruction level parallelism) that is otherwise impossible because of possible memory aliases and show that significant performance improvements are possible with this technique, another indication for the potential of run-time alias information.

7 Conclusions and Future Work

In this paper we have shown that for large C programs, the impact of using `restrict` annotations is usually limited. While those annotations can dramatically improve performance of small kernels, larger applications do not significantly benefit from even optimistic `restrict` annotations. Since the potential benefits are meager but the potential for introducing errors high, we conclude that programmer-specified non-aliasing is a bad idea in general for improving program performance.

An alternative way of enabling compilers to perform aggressive optimization even when static point analyses indicate it might be unsafe to do so, is dynamic alias analysis. In those cases in which noticeable performance improvements are theoretically possible, the run-time cost for guard code that ensures soundness of the annotations becomes important. Exploring ways to make these guards cheap, possibly by special (simple) hardware support and making the automatic generation of `restrict` more efficient, are interesting areas for future research. Moreover, we are planning on integrating the dynamic alias information directly into the optimizer thereby obviating the need for `restrict` annotations completely.

References

[1] Alex Aiken, Jeffrey S. Foster, John Kodumal, and Tachio Terauchi. Checking and inferring local non-aliasing. In *Proceedings of the 2003 ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 129–140. ACM Press, 2003.

[2] Lars O. Andersen. *Program Analysis and Specialization for the C Programming Language*. Ph.D. dissertation, University of Copenhagen, DIKU, May 1994.

[3] David Bernstein, Doron Cohen, and Dror E. Maydan. Dynamic memory disambiguation for array references. In *Proceedings of the 27th Annual International Symposium on Microarchitecture*, pages 105–111, San Jose, CA, USA, November 1994.

[4] Manuvir Das. Unification-based pointer analysis with directional assignments. In *Proceedings of the 2000 ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 35–46, Vancouver, BC, Canada, June 2000.

[5] Manuvir Das, Ben Liblit, Manuel Fähndrich, and Jakob Rehof. Estimating the impact of scalable pointer analysis on optimization. In *SAS '01: The 8th International Static Analysis Symposium*, Lecture Notes in Computer Science, Paris, France, July 16–18 2001. Springer-Verlag.

[6] Jack W. Davidson and Sanjay Jinturkar. Improving instruction-level parallelism by loop unrolling and dynamic memory disambiguation. In *Proceedings of the 28th Annual International Symposium on Microarchitecture*, pages 125–132. IEEE Computer Society Press, 1995.

[7] Jeffrey Dean and Craig Chambers. Towards better inlining decisions using inlining trials. In *Proceedings of the 1994 ACM conference on LISP and functional programming*, pages 273–282, 1994.

[8] Manel Fernandez and Roger Espasa. Speculative alias analysis for executable code. In *Proceedings of the 2002 International Conference on Parallel Architectures and Compilation Techniques (PACT '02)*, pages 222–231, Charlottesville, Virginia, September 2002.

[9] Jeffrey S. Foster and Alex Aiken. Checking Programmer-Specified Non-Aliasing. Technical Report UCB//CSD-01-1160, University of California, Berkeley, October 2001.

[10] Rakesh Ghiya, D. Lavery, and D. Sehr. On the importance of points-to analysis and other memory disambiguation methods for C programs. In *Proceedings of the 2001 ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 47–58, Snowbird, UT, USA, June 2001.

[11] A. S. Huang, G. Slavenburg, and J. P. Shen. Speculative disambiguation: a compilation technique for dynamic memory disambiguation. In *Proceedings of the 21st Annual International Symposium on Computer Architecture*, pages 200–210. IEEE Computer Society Press, 1994.

[12] Intel Corporation. *Intel Itanium Architecture Software Developers Manual*, 2002.

[13] International Organization for Standardization, Geneva, Switzerland. *ISO/IEC 9899:1999 Programming languages – C*, 12 1999.

[14] David Koes, Mihai Budiu, Girish Venkataramani, and Seth Copen Goldstein. Programmer specified pointer independence. Technical Report CMU-CS-03-128, Carnegie Mellon University, April 2003.

[15] David Koes, Mihai Budiu, Girish Venkataramani, and Seth Copen Goldstein. Programmer specified pointer independence. In *Proceedings of 2nd ACM SIGPLAN Workhop on Memory System Performance, MSP 04*, June 2004.

[16] William Landi and Barbara G. Ryder. A safe approximate algorithm for interprocedural pointer aliasing. In *Proceedings of the ACM '92 SIGPLAN Conference on Programming Language Design and Implementation*, pages 235–248, San Francisco, CA, USA, June 1992.

[17] Donglin Liang and Mary Jean Harrold. Efficient points-to analysis for whole-program analysis. In *Proceedings of the 7th European Software Engineering Conference and ACM Symposium on the Foundations of Software Engineering*, pages 199–215, Toulouse, France, September 1999.

[18] Markus Mock. *Automating Selective Dynamic Compilation*. Ph.D. dissertation, University of Washington, Department of Computer Science & Engineering, August 2002.

[19] Markus Mock, Manuvir Das, Craig Chambers, and Susan J. Eggers. Dynamic points-to sets: A comparison with static analyses and potential applications in program understanding and optimization. In *Proceedings of the 2001 ACM SIGPLAN-SIGSOFT Workshop on Program Analysis for Software Tools and Engineering*, pages 66–72, Snowbird, UT, USA, June 2001.

[20] Hemant D. Pande, William Landi, and Barbara G. Ryder. Interprocedural def-use associations for C systems with single level pointers. *IEEE Transactions on Software Engineering*, 20(5):385–403, May 1994.

[21] Matthew Postiff, David Greene, and Trevor Mudge. The store-load address table and speculative register promotion. In *Proceedings of the 33rd Annual IEEE/ACM International Symposium on Microarchitecture (Micro-33)*, pages 235–244, Los Alamitos, CA, December 10–13 2000. IEEE Computer Society.

[22] Barbara G. Ryder, William Landi, Philip A. Stocks, Sean Zhang, , and Rita Altucher. A schema for interprocedural side effect analysis with pointer aliasing. *ACM Transactions on Programming Languages and Systems*, 23(2):105–186, March 2001.

[23] Bjarne Steensgaard. Points-to analysis in almost linear time. In *Proceedings of the 23rd ACM Symposium on Principles of Programming Languages*, pages 32–41, St. Petersburg, FL, USA, January 1996.

On the Scheduling of Real-Time Heterogeneous Multiprocessor Systems-On-a-Chip

Rodrigo Santos
Dep. Ing. Eléctrica y Computadoras, Universidad Nacional del Sur, CONICET
Avda. Alem 1253, Bahía Blanca, Argentina
ierms@criba.edu.ar

and

Jorge Santos
Dep. Ing. Eléctrica y Computadoras, Universidad Nacional del Sur.
Avda. Alem 1253, Bahía Blanca, Argentina.
iesantos@criba.edu.ar

and

Ariel Fernández
Dep. Electrónica, FRBB, Universidad Tecnológica Nacional.
11 de abril 447, Bahía Blanca, Argentina.
arifer@infovia.com.ar

Abstract

In this paper a method for the scheduling of real-time tasks on heterogeneous multiprocessor systems-on-a-chip for embedded applications is presented. It is based on the partition of tasks in subtasks related by precedence and executed in different processors. The processors are linked by a common bus and therefore no delays due to interprocessors network communications are present. An extensive experimental evaluation is presented and the method is compared to other solutions recently proposed.

Keywords: Real-Time Systems, Scheduling, Embedded Systems

Resumen

En este trabajo se presenta un método para la planificación de tareas de tiempo real en sistemas multitarea-multiprocesador integrados para aplicaciones embebidas. Se basa en la partición de subtareas relacionadas por precedencias que ejecutan en distintos procesadores. Los procesadores están comunicados por un canal común y por lo tanto no hay demorás en la transmisión de mensajes asociadas a la red. Se presenta una evaluación experimental comparativa con otras soluciones propuestas recientemente en la literatura.

Palabras Claves: Sistemas de Tiempo Real, Planificación, Sistemas embebidos

1 INTRODUCTION

The development of new chips, often called *system chips* because they contain full systems, has been made possible by increased integrated circuit yields [11]. As a consequence, the last decade has seen important advances in the development of specialized multiprocessor systems-on-a-chip used in embedded applications in cars, trains, space probes, alternative energy sources, networks, mobile computing, appliances, etc. The increased integration and speed of hardware and the low consumption of energy make these kind of devices very attractive. On top of that, they can be reconfigured very easily making the design process more efficient.

Field Programmable Gate Arrays (FPGA) can be used in the implementation of embedded systems since they allow the incorporation of many functions in quite a simple way. However, their use is limited by some inherent characteristics like the cost per unit, the speed of operation, the energy consumption, the interface with the external world, etc. Another problem with FPGA is that their reuse in other applications different from the original one is not as simple as changing the flash ROM of a microprocessor.

Traditional micro-controllers have improved their performance through the use of increased number of pipelines, increased number of stages within the pipeline, bigger and faster cache memories or simply by speeding up their frequencies. The first two possibilities are not very appropriate in certain cases of hard real-time systems because they introduce an important degree of uncertainty in the determination of the worst case execution times. The last one implies more energy consumption, a fact very often not acceptable.

Very Large Instruction Word (VLIW) processors, used for instance by Texas Instruments, are another option for embedded systems. The main idea in this architecture is to process several instructions in parallel. To do this, the processors have specialized ALUs, each one connected to a data bus and a program bus. Some algorithms, *e.g.* Fast Fourier Transform (FFT), are well suited for this kind of parallelism. However, since they are very specialized, more general applications require their combination with general purpose microprocessors. This leads to the integration on a chip of multiple microprocessors linked by a common bus with access to shared RAM, ROM and I/O peripherals. An example of this kind of chips is the SMJ320C80 from Texas instruments. It has a 32-bit RISC processor with four 32 bits Digital Signals Processors (DSP) in parallel [2]. This chip was developed to work as a dedicated DSP in video and sound applications, for example noise cancellation in sonar systems. Another example is the Janus system [9]. It has two processors and it was designed to work in the power train systems of internal combustion engines. Its development was guided by the need to provide an efficient control of the fuel injection to maximize torque, minimize pollution from combustions emissions and reduce fuel consumption.

The challenge to the real-time application designer is how to exploit the increased capabilities of those systems. In this search, the scheduling of sets of real-time tasks plays a central role. The system can be analyzed in a traditional way with tasks assigned to processors following some heuristic restricted by the system constraints [13]. This assignation problem is NP-complete an has no solution in polynomial time. However, in embedded systems, the tasks are usually constrained to execute in the processors in a certain order. In fact, they can be seen as composed of several subtasks related by precedence. These subtasks, can be scheduled in the processors according to different scheduling policies like Earliest Deadline First, Rate Monotonic, Deadline Monotonic and even Round Robin. In general, the execution of subtasks in DSPs is non-preemptable so the DSP looks like a critical section in the sense used in [14]. In [3, 4] Rajkumar introduced the Multiple Priority Ceiling Protocol (MPCP) and the Distributed Priority Ceiling Protocol (DPCP) for the scheduling of tasks with shared resources in generic multiprocessors systems. Because of its generality that does not take into account the particularities of the considered architecture, the method is sometimes quite pessimistic.

In [5] a co-scheduling approach is introduced. Many tasks share multiple resources. The basic idea consists in dividing the tasks in chunks and to assign to them partial deadlines in such a way that the whole system is schedulable. In the paper, the authors propose this mechanism for the scheduling of a processor and its disk controller.

In [1], the authors improve the analysis proposed by Rajkumar by assembling two queues. In one of them, the tasks that use the dedicated processor wait till they have the opportunity to execute. In the other one, tasks that do not use the dedicated processor are enqueued until they gain access to it. They use the Hyperbolic Bound to test the schedulability of the system [6].

In this paper, a method based on the partition of the tasks in subtasks related by precedence is presented. Successive subtasks are executed in different processors. The release times and deadlines are adjusted in such a way that the release of each subtask precedes its deadline and this, in turn, precedes the release of its successor. The Earliest Deadline First policy, EDF, is used for the schedulability of the main processor and the Deadline Monotonic policy, DM, for the auxiliary ones. The method is compartively evaluated against other scheduling methods.

The rest of the paper is organized in the following way. In Section 2 the model of the two processors system is presented. In Section 3 related work is discussed. In Section 4 the Precedences Method is introduced and formalized. In Section 5 the extension for more than two processors is presented. In Section 6 the method is compared to previous solutions and finally in Section 7 conclusions are drawn.

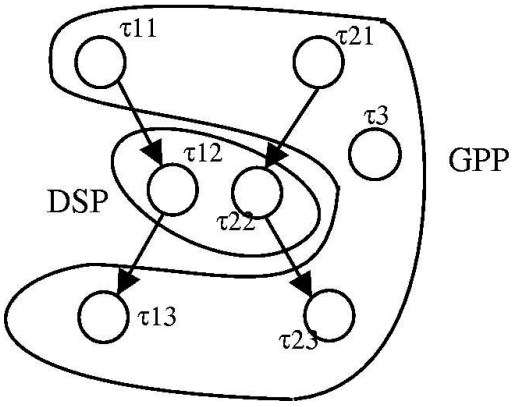


Figure 1: Architecture of the system

2 SYSTEM MODEL

In this paper, an architecture of heterogeneous multiprocessors on-a-chip is considered. The processors share the memories and the I/O devices and are linked by a common bus so that no network delays due to interprocessor communications are present.

Only two processors are considered first. One of them will be a General Purpose Processor (GPP) and the other one a dedicated processor, for instance a Digital Signal processor (DSP). Later this limitation will be relaxed and the general scheduling policy will be stated for more than two processors. Tasks running on the DSP are non-preemptable. A Remote Procedure Call (RPC) paradigm will be used to implement the communications among subtasks running on different processors. The RPC is issued by the GPP. Since its real-time kernel has to prevent a task from making an RPC if the DSP is busy, a waiting queue for the DSP is generated by the kernel to hold the tasks while they wait to execute in the DSP.

Tasks are periodic and independent. $S(m) = \{\tau_i\} = \{C_i, T_i, D_i\}$Denotes the set of tasks τ_i, $i = 1, 2, \ldots, m$.

Each task τ_i can be partitioned in subtasks τ_{ij}. $S^*(m) = \{\tau_{ij}\}$ denotes the set of subtasks τ_{ij}, $i = 1, 2, \ldots, m$, $j = 1, 2, 3$, in the case of a two processors system. The subtasks are ordered by precedence relation, $\tau_{i1} \succ \tau_{i2} \succ \tau_{i3}$. C_{ij} denotes the worst case execution time of the subtask j of task i. If $C_{i2} = 0$, the task is said to be GPP-only, which means that it doesn´t make any RPC to the DSP and can be executed without being blocked by it. If $C_{i2} > 0$, the task is said to be a DSP task.

The utilization factors for the main processor and the auxiliary one are computed in the following way:

$$U_{GPP} = \sum_{i=1}^{m} \frac{C_{i1} + C_{i3}}{T_i} \qquad U_{DSP} = \sum_{i=1}^{m} \frac{C_{i2}}{T_i} \tag{1}$$

In Figure 1 the architecture of a two processors system with three tasks is depicted. Nodes in the graph represent the subtasks, notated τ_{ij}, and the directed arcs represent precedence relations. Two of the tasks are DSP and one is GPP-only

3 PREVIOUS WORK

The problem of scheduling multiprocessors on different chips has been studied in many papers. When tasks constrained by precedence relations are executed in different processors, the communication delays and the schedulability of the interprocessor network, a real-time subsystem itself, have to be taken into consideration. In the case of processors integrated on a chip, this problem does not exist because the different processors sharing the RAM have immediate access to data produced by other processors.

Since the subtask running in the DSP is non preemptable, the processor can be considered as a critical section in the sense use in [14]. The system can be analyzed following the proposal of Rajkumar *et al* with the DPCP in [3, 4]. In this case, the schedulability test is:

$$\forall i = 1, \ldots, m \sum_{j=1}^{i-1} \frac{C_{j1} + C_{j3} + C_{j2}}{T_j} + \frac{C_{i1} + C_{i3} + C_{i2} + B_i}{T_i} \leq i(2^{1/i} - 1) \tag{2}$$

where B_i is computed as:

$$B_i \begin{cases} \max\limits_{p_j<p_i}\{C_{j2}\} + \sum\limits_{p_j>p_i} \left\lceil \frac{T_i}{T_j} \right\rceil C_{j2} & for\ a\ DSP\ task \\ 0 & for\ a\ GPP-only\ task \end{cases} \quad (3)$$

This approximation is not convenient because the execution time in the DSP affects every task, even those that do not use the DSP. The bound is therefore quite pessimistic and many feasible systems are deemed to be non-schedulable by this method.

In [1], the proposal of Rajkumar is improved by assembling two scheduling queues, one for the DSP tasks and the other one for the GPP-only ones. The requests to use the DSP are implemented by means of a blocking primitive that suspends the task in a waiting queue. The waiting task with higher priority is activated by the DSP once it finishes the execution of the current task. GPP-only tasks run independently of the DSP whenever they are ready and have priority enough to access the GPP. In this way, only the tasks that use the auxiliary processor can be blocked by lower priority tasks that use it too.

The blocking time of a task can be computed considering the longest execution time of the lower priority tasks plus the execution of all the higher priority ones that may run in one period of the task under consideration. So B_i is computed as:

$$B_i \begin{cases} C_{i2} + \max\limits_{p_j<p_i}\{C_{j2}\} + \sum\limits_{p_j>p_i} \left\lceil \frac{T_i}{T_j} \right\rceil C_{j2} & for\ a\ DSP\ task \\ 0 & for\ a\ GPP-only\ task \end{cases} \quad (4)$$

Once the blocking time for each task is calculated the schedulability can be tested with the traditional bound of Liu and Layland [7], by means of the Hyperbolic Bound (HB) [6] or by means of the Response Time Analysis (RTA) [10].

This approximation improves the possibilities of differentiating feasible from unfeasible systems because the blocking on the DSP is considered only for tasks that use it. It still has the limitation that, although it is a two- processors system, the test is done as if only one processor were available.

In [12] two scheduling methods for the synchronization of processes in distributed systems are presented. The time that a process is waiting for a response from another process is called *external blocking*. A scheduling algorithm based on the RTA divides the tasks in two independent parts, before and after the external blocking. While the first part has a release time equal to 0, the second one has a release jitter which is equal to the external blocking. Although the proposition provides an interesting way to analyse the feasibility of the system, the schedulability of the second processor is not discussed.

4 THE PRECEDENCES METHOD

The Precedences Method (PM) introduced in this paper to analyze the feasibility of the system divides each task in subtasks ordered by precedence relations. A successor subtask cannot be executed until data produced by its predecessor is available. In what follows it will be assumed that this takes place when the predecessor finishes its execution. The release times and deadlines of each subtask are selected in such a way that the precedence related subtasks can be analyzed as independent ones. In order to do that, for each τ_{ij} a release time r_{ij} and a deadline d_{ij} must be defined in such a way that:

$$r_i = r_{ij} < d_{ij} \leq r_{i(j+1)} < d_{i(j+1)} \leq r_{i(j+2)} < d_{i(j+2)},\ d_{i(j+2)} = D_i \leq T_i \quad (5)$$

It is said then that the system is coherent. The first and the third subtasks execute in the GPP while the second one executes in the DSP. Thus, the schedulability test has to be done for each processor. The EDF policy is used in the GPP and the DM policy in the DSP. As in [1], a DSP task that cannot access the DSP is blocked and waits in a special queue until it has priority enough (shortest relative deadline) to gain access. When the task that is actually running on the DSP finishes its execution, it awakes the highest priority task present in the waiting queue. Since the scheduling in this auxiliary processor is non-preemptable, it is necessary to bound the amount of time that a task may remain in the waiting queue.

The scheduling of the DSP can be established by means of the following:

Theorem 1: A set of periodic, independent, non-preemptable tasks $S(m)$ is DM schedulable if and only if:

$$\forall i, \qquad D_i \geq least\ t | t = C_i + \max_{h=i+1,\ldots,n}\{C_h\} + \sum_{j=1}^{i-1} C_j \left\lceil \frac{t}{T_j} \right\rceil \quad (6)$$

Proof: Each task meets its time constraint if and only if before its deadline it finds time enough to execute itself (first term), to withstand the blocking of tasks of lesser priority (second term) and to give way to task of higher priority (third term).□

To apply Theorem 1 to the determination of the feasibility of the auxiliary processor, it is necessary to establish an appropriate deadline for each subtask. Since in the main processor the tasks are executed following EDF, the latest moment in which a subtask running on the DSP can finish its execution and still leave enough time for the last subtask of the task to end before its deadline is given by:

$$d_{i2} = D_i - C_{i3} \tag{7}$$

Once the deadlines are established for the subtasks that run on the DSP, they are ordered by increasing values and the schedulability test following Theorem 1 is made. If the DSP is not schedulable, the system is not schedulable. If the DSP is schedulable, it is necessary to test the feasibility of the GPP.

In the schedulability analysis of the GPP, the precedence relations and the different releases and deadlines of the subtasks must be taken into account. In [8] a sufficient condition for the schedulability by EDF in systems where the tasks releases and deadlines are not synchronous with their periods is given: $S(m)$ is schedulable by EDF if $\forall h = 1, \ldots, m$, $\forall g = 1, \ldots, m$, such that $r_h \leq r_g$, $d_h \leq d_g$,

$$\sum_{r_k \leq r_g, d_k \leq d_h} C_k \leq d_h - r_g \tag{8}$$

Based on this result it is possible to study the feasibility of the GPP. It is necessary to determine for each one of the subtasks running on it, the release times, $r_{i,1}$ and $r_{i,3}$, and the deadlines, d_{i1} and d_{i3}. Establishing the deadline of the last subtask is simple because it has to be equal to the deadline of the task.

For the first subtask, the deadline is computed from the scheduling condition of the second subtask. In fact the deadline for the first subtask is the latest instant at which the second part has to be ready for execution in order to meet its deadline.

$$\forall i = 1, 2, ..., m \;\; d_{i1} = d_{i2} - least \quad t | t = C_{i2} + \max_{h=i+1,\ldots,m} \{C_{h2}\} + \sum_{j=1}^{i-1} C_{j2} \left\lceil \frac{t}{T_j} \right\rceil \tag{9}$$

For the release times of each subtask the earlier instant at which it may be ready for execution is chosen. In the case of the first subtask, it is the release time of the task. r_{i2} and r_{i3} are computed in the following way:

$$\forall i = 1, \ldots, m \;\; \forall j = 2, 3 \qquad r_{ij} = r_{i(j-1)} + C_{i(j-1)} \tag{10}$$

Once the release times and deadlines are established, the schedulability conditions can be tested. if all the subsets are schedulable the set $S(m)$ is schedulable.

The complexity of the algorithm is $O(mT_m)$ for the auxiliary processor and $O(p^2)$for the main one, where p is the number of tasks after the transformation.

5 EXPERIMENTAL EVALUATION

In this section the results of extensive simulations made to compare the Precedences Method introduced in the previous section with approaches proposed by other authors are presented. In the first set of simulations, the same specifications established in [1], although limiting the number of tasks to 10, were used, namely :

- Tasks´ periods were chosen randomly between 10 and 1000.
- Execution times for the tasks were selected in such a way that the total utilization factor of the system (GPP+DSP) fall in the interval (0.01, 0.99].
- DSP tasks represented, on the average, 80% of the total.
- The execution time of the task in the DSP was selected to be between 10% and 80% of the total execution time of the task.

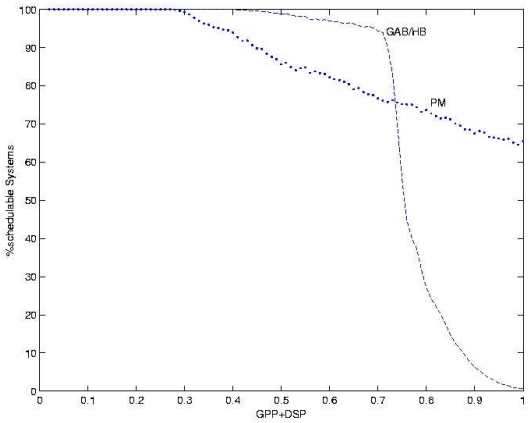


Figure 2: Percentage difference of schedulable sets by each methods

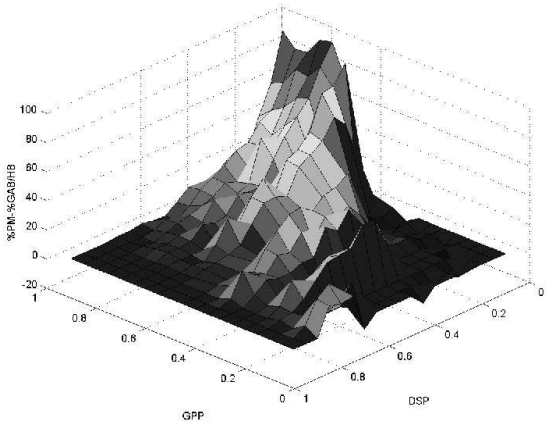


Figure 3: Difference between percentages of systems schedulable by PM and GAB/HB.

With this conditions 10^6 sets of tasks were generated and their schedulability tested by the method proposed in [1] by Gai, Abeni and Buttazzo using the Hyperbolic Bound (GAB/HB), and by the Precedences Method (PM). The percentages of the total number of sets that each method detect as schedulable were determined and their difference plotted *vs* the total utilization factor in Figure 2.

As can be seen up to an utilization factor of 0.3, the two methods produce similar results. In the interval [0.3, 0.75] GAB/HB works better but from 0.75 on, PM outperforms it.

However, the previous results were obtained under the rather restricting condition that the total utilization factor is, at most, 1, disregarding the fact that two processors on the chip allow higher total utilization factors. Thus, a second set of simulations were prepared varying both GPP and DSP utilization factors between 0.01 and 0.99. For each pair (U_{GPP}, U_{DSP}), one hundred systems were randomly generated with periods selected between 10 and 1000. In Figure 3, the difference between the percentage of systems detected as schedulable by each method was plotted for each pair.

As can be seen, as the GPP´s utilization factor approaches 1 and the utilization factor in the DSP is close to 0, the Precedences Method detects a higher number of schedulable systems. This is simply explained by the fact that GAB/HB works with a fixed priorities policy which is not good at high utilization factors, whereas PM uses EDF. As the DSP´s utilization factor is incremented, the difference is reduced to approximately 25% when the DSP is around 0.3. As both utilization factors decrease towards 0, the difference between the two methods is not significant.

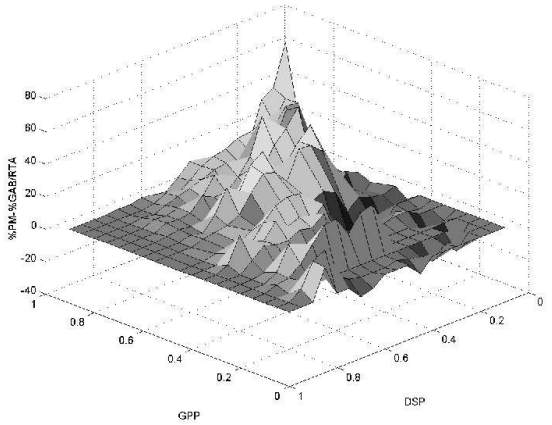


Figure 4: Difference between percentages of systems scheduled with PM and GAB/RTA

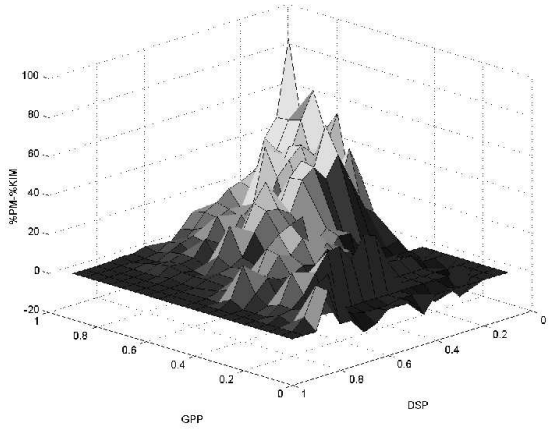


Figure 5: Difference between percentages of systems scheduled with PM and Kim´s method.

In Figure 4 the comparison is made between the PM and GAB using RTA instead of HB. Since it gives a necessary and sufficient condition for the schedulability of the system, it is natural that the difference between the two approaches is lower than before. However, in high CPU utilization factors there is still a clear advantage in favour of the Precedences approach.

Figure 5 shows the difference between the percentage of systems schedulable by PM and by the first method proposed by Kim. In this case, the release jitter was considered to be equal to the B term computed as in [1]. As can be seen, the result is quite similar to the one obtained with GAB/ RTA. As previously, there is a clear cut in favour of PM whenever the utilization factor is high.

6 GENERALIZATION TO MULTIPLE PROCESSORS ON A CHIP

In this section the generalization of the previous results is described. The method proposed up to this point is capable of analyzing the schedulability of two processors on a chip. However, it is possible that more than two processors are integrated on a chip (*e.g.* the Texas Instruments SMJ320C80) and that the tasks execute in them on a predefined order. In such cases it is quite straightforward to generalize the method. The algorithm for doing so is presented in Algorithm 1.

Algorithm 1 Schedulability Analysis for more than 2 processors

1. Divide each task in subtasks and allocate them to the proper processor.
2. Deadlines to the different subtasks are computed as follows:
   (a) The last subtask has the deadline of the task.
   (b) If the subtask´s successor is allocated to a non-preemptable processor then:
   $$\forall i,j \qquad d_{ij} = d_{ij+1} - least \quad t | t = C_{ij+1} + \max_{h=i+1,\ldots,n} \{C_{h2}\} + \sum_{g=1}^{i-1} C_g \left\lceil \frac{t}{T_g} \right\rceil$$
   (c) If the subtask´s successor is allocated to a preemptable processor then:
   $\forall i,j \qquad d_{ij} = d_{ij+1} - C_{ij+1}$
   (d) In the case that a subtask has more than one successor, deadlines are computed following cases (b) and (c) and the minimum is chosen.
3. Release times of the different tasks are computed as follows:
   (a) The first subtask has release time equal to the task release time.
   (b) If the subtask has a predecessor:
   $\forall i,j \qquad r_{ij} = r_{ij-1} + C_{ij-1}$
   (c) If the subtask has more than one predecessor, the greatest release time is chosen from the possibilities computed following (b).
4. To test the schedulability of the system, each processor has to pass the scheduling test. In the case of a non-preemptable processor, Theorem 1 should be applied. In the case of a preemptable processor condition 8 should be used.

7 CONCLUSIONS

The new trend in embedded systems with multiprocessors on a chip requires the use of new scheduling paradigms. The Precedences Method presented in this paper allows a simple implementation of high utilization factor in the different processors.

Earliest Deadline First and Non Preemptable Deadline Monotonic scheduling policies for the different processors are simple to implement. The computation of the release times and deadlines for each subtask is quite straightforward and the generalization to more than two processors is quite natural.

PM was simulated and tested against other methods proposed in the literature for the case of two processors. The simulations show a clear advantage of PM when the GPP utilization factor is high. Although the Kim´s and GAB/RTA methods work quite well in all the range of utilization factors, the paradigm proposed here outperforms them for utilization factors over 0.7.

References

[1] Gai, P., L. Abeni, G. Butazzo. Multiprocessor DSP Scheduling in System-on-a-chip Architectures. *Proc. 14th Euromicro Conference on Real Time Systems*, pp. 231-238, Viena, 2002.

[2] Texas Instruments. Military semiconductor products fact sheet SM320C80/SMJ20C80/5962-9679101 SGYV006C. Agosto 2000.

[3] Rajkumar, R., L. Sha, J. P. Lehoczky. Real time synchronization protocols for multiprocessors. *Proc. IEEE Real Time Systems Symposium*, 1988.

[4] Rajkumar, R. Synchronization in Real Time Systems: A priority inheritance approach. *Kluwer Academic Publishers*, 1991.

[5] Saewong, S., R. Rajkumar. Cooperative scheduling of multiple resources. *Proc. IEEE Real-Time Systems Symposium*, Diciembre 1999.

[6] Bini, E., G. Buttazzo, G. Buttazzo. A hyperbolic bound for the rate monotonic algorithm. *Proc. 13th Euromicro Conference on Real-Time Systems*, junio, 2001.

[7] C. L. Liu & J. W. Layland, Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment. *Journal ACM*, 20 (1), 1973, 46-61.

[8] Chetto, H., M. Silly-Chetto, T. Bouchentouf. Dynamic scheduling of real-time tasks under precedence constraints. *The Journal of Real-Time Systems*, 2, 181-194, 1990.

[9] Ferrari, A., S. Garue, M. Peri, S. Pezzini, L. Valsecchi, F. Andreatta, W. Nesci. The design and implementation of a dual-core platform for power-train systems. *Convergence 2000*, Detroit, Octubre 2000.

[10] Wellings, A., M. Richardson, A. Burns, N. Audsley, K. Tindell. Applying new scheduling theory to static priority preemptive scheduling. Report RTRS 1921129, Dep. of Computer Science, University of York.

[11] Gupta, R. Driving research in system-chip design technology. *Computer*, 36, 7, 95-97, July 2003.

[12] Kim, I-G., K-H. Choi, S-K. Park, D-Y. Kim, M-P. Hong. Real-time scheduling of tasks that contain the external blocking intervals", 2nd International Workshop on Real-Time Computing Systems and Applications, pp. 54-59, October 25 - 27, 1995 Tokyo, Japan.

[13] Santos J., E. Ferro, J. Orozco, R. Cayssials. A heuristic approach to the multitask-multiprocessor assignment problem using the empty-slots method and rate monotonic scheduling. *Real Time Systems*, 13, pp. 177-199, 1997.

[14] Sha L., R. Rajkumar, J. P. Lehoczky. Priority inheritance protocols: an approach to real-time synchronization. *Transactions on Computers*, 39, 9, pp. 1175-1185, 1990.

Identificación de Usuarios Basado en el Reconocimiento de Patrones de Tecleo

Daniel Acevedo
Universidad Central de Venezuela, Facultad de Ciencias,
Escuela de Computación, Caracas, Venezuela
dacevedo@acmgrp.com

and

Glemarys Hernández
Universidad Central de Venezuela, Facultad de Ciencias,
Escuela de Computación, Caracas, Venezuela
glema@cantv.net

and

Eugenio G. Scalise P.
Universidad Central de Venezuela, Facultad de Ciencias,
Escuela de Computación, Centro ISYS, Caracas, Venezuela
escalise@acm.org

Resumen

En este trabajo se plantea un método para la identificación de usuarios basado en el reconocimiento patrones de tecleo utilizando una Red de Base Radial. Para la realización de las pruebas de reconocimiento se tomaron datos generados por los eventos de teclado de una aplicación de mensajería instantánea por Internet. Durante el entrenamiento del modelo se utilizaron datos de tecleo de diecisiete usuarios de habla hispana. Tales datos están conformados por el tiempo transcurrido entre pares de letras tecleadas consecutivamente y el par de letra tecleado por el usuario. Estos pares fueron tomados de una lista de cuarenta pares seleccionados durante el estudio. Como resultado se obtuvo un módulo de reconocimiento de patrones de tecleo con resultados de reconocimiento aceptables.

Palabras claves: inteligencia artificial, redes neuronales, redes neuronales RBF, patrones de tecleo, biometría, identificación de usuarios.

Abstract

In this work it is presented a method for the user identification based on the pattern recognition of keystrokes, using a Radial Base Neural Network. For the accomplishment of the recognition tests it were used data generated by the events from keyboard of an instant messaging program. During the training of the model, it were used keystroke data of seventeen hispanic speech users. Such data contains the time passed between pairs of consecutively letters and the pair of letters keyed in by the user. These pairs were taken from a list of forty pairs selected during the study. As a result, we have obtained a recognition module of keystrokes with acceptable recognition levels.

Keywords: artificial intelligence, neural networks, radial basis function networks, keystroke pattern, biometry, user recognition.

Introducción

Desde tiempos inmemoriales, la información ha jugado un papel importante para el hombre. Esto se evidencia en los diferentes métodos utilizados para asegurar su confidencialidad. Durante el antiguo Egipto, la escritura no era accesible al común de la gente, garantizando que sólo un grupo reducido de personas pudieran acceder a la información. Posteriormente, el ejército romano utilizó el cifrado del Emperador Julio Cesar para ocultar sus mensajes, desplazando el alfabeto tres letras, siendo imposible descifrar el mensaje sin tener conocimiento de ello. En la Segunda Guerra Mundial, Churchill y Roosevelt se comunicaban telefónicamente sólo si estaban seguros de que no lo estuvieran espiando, o que si lo hacían no los entenderían. [8]

La información es sinónimo de poder. Resulta fácil imaginar lo que podría suceder si se conoce toda la información que maneja la NASA, el departamento de seguridad de los EEUU o las instituciones bancarias a nivel mundial.

Por otra parte, la sociedad actual demanda cada vez mayor información y los medios utilizados para transmitirla se encuentran, generalmente, al alcance de muchas personas, lo que constituye una gran ventaja: la información se encuentra disponible para ellas. Sin embargo, no todos están dispuestos a compartir la información que poseen, por lo que es necesario implementar mecanismos de seguridad que limiten el acceso sólo a personas autorizadas. De igual manera, se debe garantizar que la información proviene de fuentes confiables debido al valor asociado a ésta.

Es por ello que en este artículo se plantea una alternativa biométrica para la identificación de usuarios que contribuye a incrementar la seguridad de los sistemas. El mismo se estructura de la siguiente manera: se presenta el planteamiento del problema, luego se expone la técnica de reconocimiento de patrones utilizada en la identificación de usuarios, junto con los resultados obtenidos, finalizando con las conclusiones de la investigación.

1 Planteamiento del problema

La información es un bien que adquiere cada vez mayor relevancia. No solo debe ser protegida de posibles ataques y limitar el acceso a ella, sino además, se debe garantizar que proviene de fuentes confiables.

La identificación del usuario o cliente es un punto fundamental para garantizar la seguridad en los sistemas de información. Un ejemplo de ello puede observarse en las instituciones bancarias que solicitan a sus clientes una clave compuesta por varios dígitos para movilizar su dinero con la tarjeta de débito. Igualmente, algunos celulares necesitan de un número de identificación personal (PIN) para poder iniciar sus funciones. Acceder a la mayoría de los servicios provistos en Internet requiere de un *password* o contraseña. Los Sistemas Operativos que funcionan en ambientes locales o de red utilizan también este mecanismo. Todas estas situaciones tienen algo en común: el uso de una clave personal para acceder al sistema y poder operar en él. Esta clave personal constituye a la vez un riesgo. Al ser información que puede ser compartida o descubierta, el sistema tiende a ser vulnerable al fraude.

Por otro lado, hacer uso de Internet para intercambiar información es muy común hoy día para quienes tienen acceso al medio. Un estudio realizado en España por AIMC [3] revela que el 95.3% de los usuarios de Internet utiliza el servicio de correo electrónico, el 38.8% participa en charlas interactivas o *chats* y un 18.2% hace uso de los foros de discusión. De aquí que sea importante tener la certeza de que la información proviene realmente de quien dice provenir. Una manera de garantizarlo es a través de la identificación de los usuarios que intervienen en el intercambio.

La tecnología biométrica resulta una alternativa al momento de reconocer o identificar a un individuo, ya que utiliza las características del mismo para su identificación. De este modo, sólo el usuario es propietario de la información. Dichas características pueden ser físicas, tales como las huellas dactilares, el contorno del rostro y la retina, o aprendidas, puesto que miden el comportamiento del usuario, tales como el timbre de voz y la firma manuscrita. Sin embargo, los equipos utilizados para captar estas características resultan bastante costosos y, por consiguiente, poco disponibles para el usuario común.

Otra alternativa biométrica menos costosa está basada en la dinámica de tecleo del usuario. La misma consiste en identificar a un individuo tomando en cuenta el modo de teclear un texto, ya que se ha demostrado que cada individuo posee patrones únicos de tecleo asociados a la velocidad de tecleo o a la presión ejercida al teclear [4].

Entonces, teniendo en cuenta lo importante que es mantener la seguridad y las necesidades de tener un método alternativo de bajo costo para el reconocimiento de usuarios, se desarrolló un sistema reconocedor de patrones de tecleo [1] capaz de diferenciar a un usuario de otro, utilizando como equipo de adquisición de datos el teclado, tomando en cuenta la velocidad del usuario al teclear como característica de tecleo. Como métrica se utilizó el intervalo de tiempo registrado entre un par de letras tecleadas de manera consecutiva.

En detalle, el sistema desarrollado captura los eventos de teclado del usuario y registra los tiempos entre pulsaciones. Con esa información se construye una plantilla de reconocimiento que permite identificar unívocamente a un usuario, es decir, se determinan los patrones de tecleo asociados a éste.

Para efectuar el reconocimiento, se utilizó una de las técnicas de reconocimiento de patrones que mejor desempeño obtuvo en las pruebas que se realizaron para tal fin.

1.1 Ambiente de trabajo

Lo más importante para un sistema de identificación de usuarios es el grado de certeza de contar con un método efectivo para el reconocimiento o identificación. Es por ello que fue necesario que el sistema reconocedor de patrones de tecleo a desarrollar arrojase resultados confiables, reconociendo efectivamente a un individuo y rechazando eficazmente a los intrusos, distinguiendo entonces entre un grupo de individuos.

Un sistema que permita reconocer a un individuo por medio de la velocidad de éste al teclear puede tener múltiples aplicaciones. La facilidad de uso que proporciona al usuario y el bajo costo que representa al no necesitar equipos de avanzada tecnología, lo hacen una opción para el reconocimiento de usuarios bastante atractiva. De esta manera, el sistema a desarrollar debía ser independiente y usable, es decir, capaz de brindar las facilidades necesarias para que pudiera ser utilizado por cualquier aplicación que generara eventos de teclado. Adicionalmente, debía permitir el uso de éste en diferentes escenarios.

Para darle respuesta a cada uno de los requerimientos planteados, el sistema reconocedor de patrones de tecleo se desarrolló como un módulo independiente de reconocimiento, en el cual se implementaron las funciones pertinentes al método de reconocimiento de patrones seleccionado, y que permitiese establecer una conexión con otras aplicaciones.

Para probar el desempeño del módulo en el reconocimiento de patrones de tecleo, fue necesario diseñar e implementar una plataforma de operación donde se pudiesen realizar las pruebas y así verificar si el reconocimiento se realizó exitosamente. Dicha plataforma cuenta con los siguientes elementos:

- Un generador de eventos de teclado (AGE): Esta aplicación es la encargada de captar las interrupciones de teclado y generar los eventos de teclado del individuo en estudio. Entre los AGEs más comunes que pueden ser utilizados para el reconocimiento se tienen editores de texto, clientes de correo y *chats*.
- Un filtro-receptor (AFR): Esta aplicación monitorea todos los eventos de teclado proveniente de AGE, registra los tiempos ocurridos entre pares de teclas y filtra la información que va a ser suministrada al módulo.
- Un almacén de datos (AD): Este elemento es el responsable de almacenar las plantillas de reconocimiento de cada usuario encontradas durante la etapa de entrenamiento. Una plantilla de reconocimiento representa las características únicas del usuario utilizadas para la identificación a partir de ejemplos biométricos.

1.2 Captura de datos

Para la captura de los datos se utilizó el producto del servicio de mensajería por Internet MSN Messenger versión 4.6 como un AGE para probar el módulo de reconocimiento dada las siguientes ventajas:

1. La frecuencia de uso de este servicio por parte de sus usuarios genera gran cantidad de información de tecleo que puede ser usada en el reconocimiento de los individuos en estudio.
2. La gran cantidad de usuarios que posee el servicio facilita la búsqueda de individuos para probar la aplicación.
3. La aceptación del servicio entre sus usuarios facilita el hecho de que más individuos se sumen al estudio.
4. El producto puede descargarse gratuitamente vía Internet y su instalación resulta sencilla.
5. Cuenta con un API que facilita la obtención de los eventos generados por esta aplicación, de manera transparente para los usuarios.

Como AFR se desarrolló una aplicación, denominada TGD, que guarda la información de cada par de letras tecleadas consecutivamente y el tiempo transcurrido entre una y otra letra para todas las ventanas de conversación de MSN Messenger que el usuario mantenga con otras personas.

La validez de los datos desempeña un rol importante para la precisión del reconocimiento. Al basar el estudio en registro de tiempos, la manera en que midan los datos debe garantizar que las mediciones sean independientes a los ciclos de reloj del procesador de la máquina del usuario. Para registrar el tiempo de tecleo entre dos letras se utilizó una función tomada del API de Windows llamada `TimeGetTime` [10]. Esta función retorna el tiempo actual del sistema en milisegundos y su resultado no varía en relación a la velocidad del procesador. La misma es comúnmente utilizada en aplicaciones que demandan una alta resolución en el tiempo, tales como programas 3D, donde se cuenta la cantidad de cuadros desplegados por segundo.

Para el muestreo de los tiempos de tecleo en las ventanas de conversación de MSN Messenger se utilizó la función `SetWindowsHookEx` [10], perteneciente también del API de Windows. Esta función está disponible dentro de una librería de enlace dinámico (DLL) y captura todos los eventos de teclado generados en el ambiente para así poder tomar el tiempo transcurrido entre cada par de letras tecleadas consecutivamente. Para identificar cuáles de estos eventos ocurren dentro del ambiente de una ventana de conversación de MSN Messenger, se utilizó la función `CallWindowProc` [10], también perteneciente al API de Windows y disponible dentro de un DLL.

El resultado del monitoreo de las letras escritas en MSN Messenger y el tiempo que transcurre entre la ocurrencia de cada una de ellas es almacenada en un archivo de registro. El formato del almacenamiento de los datos en el archivo es el siguiente: al inicio del mismo se encuentra el correo del usuario que se conectó al servidor de MSN Messenger, posteriormente, en cada una de las líneas siguientes se tiene la información de cada par de teclas presionadas dentro

de una ventana de conversación y el tiempo transcurrido entre ellas.

Para detectar cuando un usuario se está conectando al servidor de MSN Messenger, se utilizó el API de MSN Messenger versión 4.6 [10]. Esta interfaz permite capturar entre otros mensajes, los eventos de conexión al servidor, junto con el correo con el que se está haciendo la autenticación del usuario.

1.3 Selección de pares

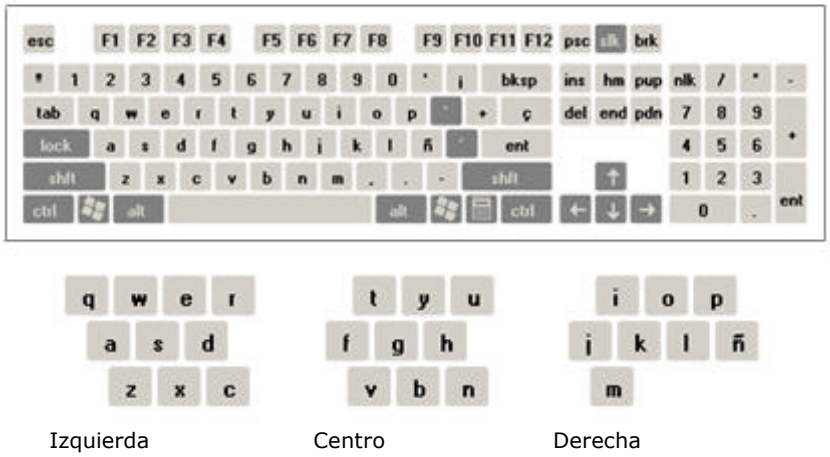
La métrica seleccionada para realizar el reconocimiento es la velocidad del usuario al teclear, teniendo en cuenta el tiempo que toma éste al pulsar dos teclas de manera consecutiva. Como es de esperarse, no fue necesario analizar toda los datos suministrados por el usuario. Si se observan los pares de letras que se pueden formar a partir del alfabeto, resulta evidente que existen combinaciones que son poco probables que ocurran al escribir un texto, tales como el par wx o el par qz, tomando en cuenta que el idioma utilizado en todas las conversaciones es en español. Adicionalmente, dependiendo del contexto en donde se esté escribiendo, se van a usar o dejar de usar ciertas palabras y con ello, ciertos pares pueden tener más o menos número de ocurrencias. Estas diferencias se observan al redactar una carta a una institución, al escribir un correo personal o al chatear con amigos.

Luego, teniendo en cuenta que se utilizó el MSN Messenger como un generador de eventos de teclado, fue necesario analizar los pares de letras con mayor número de ocurrencias entre los usuarios al utilizar esta aplicación. Para tal fin se desarrolló un programa que registra todos los eventos de teclado generados por el usuario al utilizar el MSN Messenger, al cual se le llamó TGD. Cuando se producen eventos de teclado, se registran los pares de letras asociados a los eventos ocurridos. Los eventos correspondientes a números o caracteres especiales fueron ignorados. De esta manera, se produce un archivo de registro con la información de los pares de letras tecleados consecutivamente al enviar un mensaje a algún contacto del usuario en MSN Messenger.

Es importante resaltar que la pulsación de un caracter especial, como el acento, o una tecla de control, como *shift*, entre un par de letras trae como consecuencia que dicho par no sea tomado en cuenta para el estudio, ya que las letras pertenecientes al par no fueron tecleadas de manera consecutiva.

TGD fue instalado a un grupo de treinta usuarios con el objetivo de recolectar datos referentes a los mensajes escritos por éstos al mantener una conversación. Teniendo en cuenta la frecuencia con que es utilizado Messenger entre las personas que conformaron el grupo, se estimó que el tiempo de uso de TGD fuese de dos semanas. Una vez culminado este tiempo, sólo siete usuarios tenían información relevante para el estudio y los archivos generados por estos usuarios fueron procesados con el fin de determinar los pares con mayor número de ocurrencias entre ellos.

Cada usuario generó un archivo de registro que fue tabulado y comparado con el resto de los usuarios. Para cada individuo, los pares fueron ordenados de mayor a menor de acuerdo al número de ocurrencias del par y numerados para obtener la posición de éste en cada individuo. Luego, para obtener la posición del par en el grupo, se promedió la posición que ocupaba el par en cada uno de los individuos. Esta última posición fue utilizada para seleccionar los pares que serían usados para el reconocimiento de todos los usuarios del sistema.



Izq – Izq: ES, EN, AS, ER, DE, AR, RA, RE, DA, SE, SA
Cen – Cen: UN, NT
Der – Der: LO, MO
Izq – Cen: UE, TA, TE, AN, ST, NA, ND
Cen – Der: NO, ON, TO
Izq – Der: LA, OS, CO, DO, OR, IE, AL, ME, MA, RO, EL, PA, SI, LE, AM

Figura 1. Distribución espacial de pares en el teclado.

Al analizar los pares con mayor número de ocurrencias, se evidenció que la gran mayoría de éstos se encontraban ubicados del lado izquierdo del teclado. Para que la distribución de los pares sobre el mismo fuese regular, además

del promedio ocupado en el grupo, se verificó la posición del par sobre el teclado, dividiendo el espacio ocupado por las teclas de letras en seis regiones: izquierda-izquierda, centro-centro, derecha-derecha, izquierda-centro, centro-derecha, izquierda-derecha (ver Figura 1).

Durante un análisis de factibilidad de reconocimiento de patrones de tecleo realizado previamente [AHS02], se trabajó con diecinueve pares de letras. En vista de que los primeros diecinueve pares en ese estudio no cubrían las tres regiones, se decidió aumentar progresivamente el número de pares hasta que las tres estuviesen cubiertas. De esta manera, el número de pares seleccionados para el desarrollo de la aplicación fue de cuarenta.

Los pares seleccionados para el estudio fueron los siguientes: ES, EN, UE, AS, ER, LA, DE, AR, TA, NO, RA, TE, AN, OS, CO, ST, DO, ON, OR, RE, IE, AL, ME, LO, MA, RO, EL, PA, TO, MO, UN, SI, DA, NA, SE, LE, ND, AM, SA y NT. En la Figura 1 se muestra la distribución de los pares sobre el teclado.

2 Técnica de Reconocimiento de Patrones utilizada en la Identificación de Usuarios

Una vez identificados los pares a utilizar, fue necesario determinar la técnica de reconocimiento de patrones que mejor resultado arrojara para el reconocimiento de patrones de tecleo. Las técnicas de reconocimiento seleccionadas para tal prueba fueron las siguientes: método estadístico basado en una distribución normal [6], mínima distancia [6], k-vecinos [5], perceptrón multicapa [7] y red de función de base radial [11]. Para ello, se realizaron un conjunto de pruebas con diferentes métodos, donde los datos utilizados fueron recolectados durante cuatro semanas a través del MSN Messenger y almacenados en un archivo de registro. Se analizaron un total de diecisiete archivos correspondientes a tiempos de tecleo de diecisiete usuarios distintos.

Para obtener unos resultados confiables, se realizaron un total de cinco pruebas con distintos datos del archivo de registro para cada uno de los métodos probados. Con los datos obtenidos de cada usuario, se formaron cinco particiones disjuntas. Para cada prueba, se combinaron cuatro particiones para entrenar el sistema y una para realizar el test de reconocimiento.

Los resultados arrojados por las pruebas realizadas en [2] mostraron que una red de base radial tiene un mejor desempeño que el resto de las técnicas de reconocimiento de patrones utilizadas. En lo que resta de este trabajo se mostrarán las pruebas realizadas sobre un reconocedor de patrones de tecleo que utiliza una red neuronal de base radial (RBF) para realizar el reconocimiento.

2.1 Red de Función de Base Radial (FBR)

Las redes de función de base radial (RBF) [11] se caracterizan por tener un entrenamiento híbrido, que incorpora aprendizaje supervisado y no supervisado. La arquitectura de este tipo de red es de tres capas: capa de entrada, capa oculta y capa de salida.

Las neuronas de la capa de entrada no realizan ningún tipo de procesamiento, simplemente envían los valores de entrada a la capa oculta. En la capa oculta se calcula la distancia que separa el vector de entrada de los centroides de esta capa, aplicándole luego una función gaussiana. En la capa de salida se realiza un procesamiento lineal.

El aprendizaje se realiza en dos etapas. Se entrena primero la capa oculta y luego la capa de salida. En la capa oculta se determinan los centroides y las desviaciones estándares para los vectores de entrada, mientras que en la capa de salida se determinan los valores de los pesos.

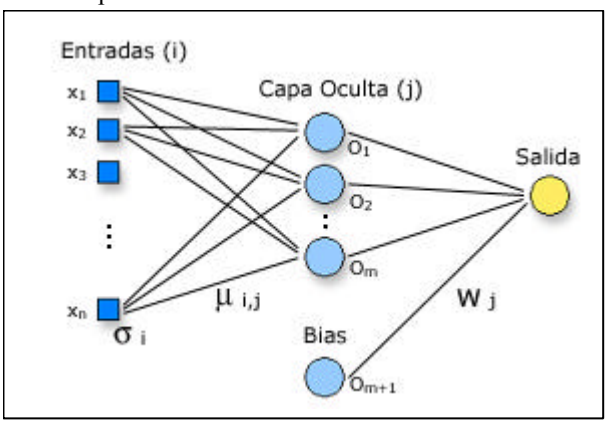


Figura 2. Arquitectura de una RBF

En la Figura 2 se puede observar la arquitectura de la red utilizada en el Módulo de Reconocimiento. Para el desarrollo del módulo, se determinaron tanto los centroides μ_{ij} por valor de entrada x_i como número de neuronas existentes en la capa oculta, utilizando para ello el algoritmo de k-medias [9]. Luego, los valores para las desviaciones estándares de cada neurona i de la capa de entrada se calcularon como la mayor de las distancias existentes entre los centroides para esa neurona. Una vez encontrados estos valores, la segunda etapa consistió en hallar los valores para las conexiones entre la capa oculta y la capa de salida que permitieran una mayor

convergencia de la red.

2.2 Experimentos y Resultados

El primer experimento realizado para la evaluación de este método se basó en la verificación de usuarios, el cual indica, a partir de valores de entrada (tiempos de tecleo) y el identificador del usuario, si efectivamente se trata del usuario que se identificó. De esta manera, existe una plantilla de reconocimiento por cada individuo del grupo. Los valores usados para determinar las medias y las desviaciones estándar fueron los tiempos de tecleo por cada par que pertenecían al individuo al cual se le iba a construir la plantilla de reconocimiento.

El número de neuronas en la capa de entrada fue de cuarenta, correspondiente a los cuarenta pares de letras seleccionados. El vector de entrada contiene tiempos de tecleo para cada par seleccionado; se utilizaron entre 12 y 30 vectores de tiempo por cada usuario. La cantidad de neuronas utilizado para la capa oculta fue de veinticinco. Al tratarse de verificación de usuarios, en la capa de salida existe una única neurona cuya salida es igual a 1 para indicar que reconoció y 0 en caso contrario.

Se realizaron un total de cinco pruebas con conjuntos disjuntos de vectores de tiempo para el entrenamiento y el reconocimiento. En la Tabla 1 se presentan los resultados de la verificación de usuarios para este método, donde la columna **IND** indica el número del individuo en estudio, las columnas Prueba i identifican el número de la prueba, la columna A representa el nivel de aceptación (total de entradas reconocidas como pertenecientes al usuario IND entre el total de tiempos del usuario IND), las columna EFA y EFR exponen el error de falsa aceptación (cuando un usuario no autorizado es aceptado por el sistema) y el error de falso rechazo (cuando un usuario autorizado es rechazado por el sistema) respectivamente, mientras que la columna PA muestra el promedio del nivel de aceptación para el individuo IND.

IND	Prueba 1			Prueba 2			Prueba 3			Prueba 4			Prueba 5			PA
	A	EFA	EFR	A	EFA	EFR	A	EFA	EFR	A	EFA	EFR	A	EFA	EFR	
1	0,64	0,05	0,36	0,55	0,1	0,45	0,64	0,07	0,36	0,73	0,12	0,27	0,73	0,06	0,27	0,65
2	0,58	0,05	0,42	0,75	0,06	0,25	0,75	0,04	0,25	0,83	0,08	0,17	0,67	0,04	0,33	0,72
3	0,7	0,07	0,3	0,7	0,05	0,3	0,43	0,05	0,57	0,7	0,07	0,3	0,78	0,06	0,22	0,66
4	0,81	0,07	0,19	0,71	0,07	0,29	0,62	0,05	0,38	0,48	0,07	0,52	0,76	0,07	0,24	0,68
5	0,88	0,04	0,12	0,85	0,03	0,15	0,79	0,04	0,21	0,82	0,05	0,18	0,67	0,04	0,33	0,8
6	0,67	0,06	0,33	0,44	0,07	0,56	0,67	0,04	0,33	0,44	0,07	0,56	0,44	0,06	0,56	0,53
7	1	0,03	0	0,67	0,07	0,33	0,5	0,05	0,5	0,5	0,07	0,5	0,75	0,06	0,25	0,68
8	0,71	0,05	0,29	0,81	0,07	0,19	0,9	0,06	0,1	0,86	0,06	0,14	0,86	0,05	0,14	0,83
9	0,88	0,02	0,12	0,92	0,03	0,08	0,92	0,04	0,08	0,84	0,01	0,16	0,8	0,01	0,2	0,87
10	0,64	0,02	0,36	0,86	0,02	0,14	0,79	0,01	0,21	0,93	0,04	0,07	0,79	0,03	0,21	0,8
11	1	0,05	0	0,8	0,05	0,2	0,87	0,07	0,13	0,8	0,04	0,2	0,6	0,05	0,4	0,81
12	0,83	0,1	0,17	0,75	0,1	0,25	0,5	0,09	0,5	0,58	0,04	0,42	0,75	0,07	0,25	0,68
13	0,88	0,08	0,12	0,88	0,08	0,12	0,94	0,03	0,06	0,88	0,06	0,12	0,71	0,03	0,29	0,86
14	0,93	0,06	0,07	0,7	0,09	0,3	0,7	0,09	0,3	0,81	0,11	0,19	0,67	0,07	0,33	0,76
15	0,67	0,04	0,33	0,89	0,03	0,11	0,67	0,03	0,33	0,78	0,03	0,22	0,72	0,05	0,28	0,74
16	0,87	0,01	0,13	0,76	0,02	0,24	0,76	0,01	0,24	0,83	0,02	0,17	0,87	0,01	0,13	0,82
17	0,67	0,07	0,33	0,79	0,06	0,21	0,67	0,07	0,33	0,85	0,09	0,15	0,76	0,07	0,24	0,75
	0,79	0,05	0,21	0,75	0,06	0,25	0,71	0,05	0,29	0,74	0,06	0,26	0,72	0,05	0,28	0,74

Tabla 1. Resultados para el RBF con 25 neuronas

Los resultados obtenidos para esta prueba muestran que el porcentaje general de reconocimiento se encuentra alrededor del 74%, lo que significa que de cada 100 intentos de reconocimiento, aproximadamente 74 serán reconocidos. El nivel de aceptación para cada individuo del grupo de estudio está por encima del 65%, excepto para el individuo 6 que posee un valor de 53% de aceptación, sin embargo la cantidad de vectores utilizados para el entrenamiento de la red de este individuo fue el menor de todo el grupo.

También se observa que los valores obtenidos para el error de falsa aceptación (EFA) no supera el 6%, lo que muestra que es capaz de rechazar un individuo no autorizado el en 94% de las veces.

Otro de los experimentos realizados consistió en comprobar la capacidad del método para rechazar intrusos, determinando el error de falsa aceptación (EFA) para un grupo de cinco usuarios. Para cada individuo de la muestra, el experimento consistió en probar el nivel de rechazo de 13 vectores de tiempos, con la salvedad de que este procedimiento se realizó un total de 16 veces, una por cada usuario del grupo de estudio, excluyendo al individuo perteneciente la muestra. De esta manera, se realizó una correspondencia de los tiempos del individuo de la muestra con las plantillas de reconocimiento del resto de los usuarios del grupo.

En la siguiente tabla se muestra la cantidad de patrones reconocidos de los individuos en estudio, por los individuos

del entrenamiento.

	IND 3	**IND 4**	**IND 6**	**IND 11**	**IND 12**
1	0	0	0	0	0
2	0	0	2	0	0
3	*	3	4	2	3
4	2	*	2	0	0
5	0	0	0	0	0
6	0	4	*	0	2
7	0	0	0	4	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0
11	0	0	0	*	0
12	0	3	2	0	*
13	0	0	0	0	0
14	0	7	5	0	3
15	0	1	1	1	0
16	0	3	3	0	0
17	6	0	1	0	0
EFA	0,0362	0,0950	0,0905	0,0317	0,0498

Tabla 2. Resultados del experimento 2

Como se puede observar en la Tabla 2, el porcentaje de falsa aceptación se encuentra alrededor del 5%, lo que indica que el método seleccionado es capaz de rechazar, con un 95% de efectividad, un intruso dentro del conjunto de entrenamiento.

El último experimento consistió en evaluar el desempeño del método al suministrar información incompleta a las plantillas de verificación, completándolas con las medias obtenidas en el entrenamiento para cada par, determinando el nivel de aceptación y el error de falso rechazo. Este experimento mide la variación entre el uso de patrones completos de usuarios y el uso de información histórica para el reconocimiento.

La muestra y los tiempos de tecleo utilizados para este experimento fueron los mismos que los del experimento anterior. Para crear el vector de tiempo de los 40 pares seleccionados, los tiempos registrados se tomaron en el orden en que fueron tecleados hasta completar 35 pares. El resto fue completado con los valores de las medias. Una vez construido el vector, era presentado a la red del individuo correspondiente para su reconocimiento. Luego, se presentó el mismo vector con las mismas características, completando los valores de las medias faltantes con valores reales del individuo.

En la tabla siguiente se muestra la relación de los resultados obtenidos al completar los valores de entrada del vector de verificación contra los vectores completados por valores de entrada del usuario.

	IND3	**IND4**	**IND6**	**IND11**	**IND12**	**TOTAL**	**%**
Número de plantillas completadas	209	170	119	187	153	838	
Reconocimiento de la plantilla completada igual al reconocimiento de la plantilla final	208	163	119	186	152	828	98,8
Reconocimiento de la plantilla completada diferente reconocimiento de la plantilla final	1	7	0	1	1	10	1,2

Tabla 3. Resultados del Experimento 4

Los resultados arrojados por este experimento muestran que para la cantidad de medias calculadas por individuo durante el reconocimiento, cinco en este caso, existe una variación en el reconocimiento del 1,2%, es decir, en el 98,8% de las veces el resultado del reconocimiento es igual completando el vector de tiempos que no haciéndolo. De esta manera, se puede utilizar este método para calcular la verificación de un individuo sin necesidad de tener el vector de reconocimiento completo utilizando información histórica del individuo al momento del entrenamiento.

3 Conclusiones

Para un escenario donde los tiempos registrados pertenecen a una aplicación de mensajería instantánea por Internet, específicamente a los cuarenta pares de letras con mayor número de ocurrencias para esa aplicación, en una muestra de diecisiete individuos, se determinó que una red neuronal basada en Funciones de Base Radial puede reconocer efectivamente a un individuo por sus características de escritura al utilizar el teclado.

El promedio de reconocimiento para un individuo fue de 74%, lo que significa que de cada 100 intentos, 74 resultarán exitosos y 26 resultarán fallidos, siendo este último el error de falso rechazo (EFR), es decir, que el 26% de las veces el individuo en cuestión será rechazado. Por otra parte, se tiene que al tratar de reconocer a un individuo con las características de tecleo pertenecientes a otro individuo de la muestra, el error de falsa aceptación (EFA) es de 5%, es decir, de cada 100 intentos, el individuo no será reconocido en 95 ocasiones.

Por otra parte, al realizar las pruebas para cada uno de los individuos, se observó que durante la etapa de entrenamiento para todos los métodos probados, el error de entrenamiento para un individuo tiende a disminuir a medida que aumenta la cantidad de información de tecleo de éste. Esto incide en la calidad del reconocimiento para cada individuo.

Los resultados obtenidos para verificar el nivel de rechazo indican que la probabilidad de reconocimiento al adicionar un individuo ajeno a la muestra es baja, pero superior al error de falso rechazo de los individuos del grupo de estudio. Esto permite observar que existe una separación de las características de los individuos de la muestra y que un individuo ajeno al estudio puede llegar a ser reconocido. En consecuencia, el método mantiene su efectividad en un entorno cerrado de individuos. Sin embargo, se pueden agregar nuevos individuos sin mayor problema agregando una nueva plantilla de reconocimiento, la cual se obtiene luego de un entrenamiento con datos del nuevo individuo.

En el último de los experimentos descritos, se pudo observar que cuando ya se tenían los valores para 35 pares de los 40 necesarios para el reconocimiento, los resultados eran muy similares a los obtenidos luego de esperar que los 40 pares fuesen completados. La diferencia era de aproximadamente un ±1.2%, por lo que se puede utilizar información histórica para completar el vector de tiempos durante el reconocimiento.

En resumen, el módulo de reconocimiento de patrones de tecleo desarrollado resulta ser una alternativa práctica, confiable y de bajo costo para la verificación de usuarios. Adicionalmente, por su característica modular (es un objeto COM) puede ser reutilizado entre diversas aplicaciones que generen eventos de teclado y requieran algún mecanismo de identificación basado en este tipo de información biométrica.

A nivel comercial, existen muy pocas aplicaciones basadas en este tipo de tecnología. Entre los productos comerciales disponibles en el mercado se destaca BioPassword de BioNet Systems [4] utilizado básicamente para reconocimiento de *passwords*. La mayoría de productos catalogados en esta categoría están basados en la captura y clasificación de información obtenida mediante teclado, pero no reconocimiento biométrico.

Referencias

[1] Daniel Acevedo y Glemarys Hernández. Desarrollo de un módulo de identificación de usuarios basado en reconocimiento de patrones de tecleo. Trabajo Especial de Grado. Escuela de Computación. Facultad de Ciencias. Universidad Central de Venezuela. Mayo 2003.

[2] Daniel Acevedo y Glemarys Hernández. *Técnicas de reconocimiento de patrones y redes neuronales artificiales (RNAs)*. Seminario de Investigación. Escuela de Computación. Facultad de Ciencias. Universidad Central de Venezuela. Octubre 2002.

[3] AIMC (Asociación para la Investigación de Medios de Comunicación). *Navegantes en la Red: Quinta encuesta AIMC a usuarios de Internet*. España: Enero, 2003. http://www.aimc.es

[4] Bionet System LLC. *Biopassword Keystroke Dinamics*. Technical Report. 2002. http://biopassword.com

[5] T.M.Cover and P.E.Hart. *Nearest Neighbor Pattern Classi?cation*. Trans.IEEE Inform.Theory, IT-13, pp21-27. 1967.

[6] K.Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, Inc. 1990.

[7] J. Hertz, A. Krogh y R.G. Palmer. *Introduction to the Theory of Neural Computation*. Addison-Wesley. 1991.

[8] Federico Kuhlmann y Antonio Alonso Concheiro. *Información y Telecomunicaciones*. Fondo de cultura Economica. México, 1997. http://www.cft.gob.mx/html/la_era/info_tel/it0.html

[9] J.B. MacQueen. *Some methods for classification and analysis of multivariate observations.* In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability. Volume I, Statistics. University of California Press. 1967.

[10] MSDN Developer Center. http://msdn.microsoft.com

[11] Mark J.L. Orr. *Introduction to Radial Basis Function Networks.* Centre for Cognitive Science. University of Edimburgh. Scotland. April 1996. http://www.anc.ed.ac.uk/~mjo/papers/intro.ps.gz

Time-Variant Watermarking of MPEG-Compressed Digital Videos

Ernst L. Leiss*
Department of Computer Science
University of Houston
Houston Texas 77204-3010

coscel@cs.uh.edu
voice 713-743-3359
fax 713-743-3335

Abstract

Watermarks allow one to embed information into digital videos in a way that is imperceptible to the viewer. This information can be used to establish ownership, trace origin of copies, and verify the integrity of the video. Watermarking may be compared to injecting additional energy; to ensure that this injection remains unnoticeable, it should be as small as possible. We outline an approach that permits a significant increase of the amount of information that can be accommodated in a watermark without any increase in the complexity of the process, namely time-variant watermarks. Since data compression is an important aspect in storing and distributing digital videos, we formulate our approach assuming the video is represented in an MPEG format. We discuss implementation issues of time-variant watermarks, with special emphasis on their advantages over the usual time-invariant watermarks. We comment on defeating attacks using filtering, cropping, resizing, and other standard methods used to defeat watermarks, such as changing existing frames, as well as new attacks, such as removing, repeating or permuting frames.

Keywords: Digital video, MPEG compression, watermarks, time-variance.

* Support of this work under NSF Grant SFS-0313880 is acknowledged.

1. Introduction and Motivation

All digital information can be copied perfectly since any string consisting of 0's and 1's is indistinguishable from its copy [Leiss, 1982]. The existence of perfect copies has numerous implications for data security and integrity; note that in the physical world, perfect copies do not exist by definition. Since it is impossible to distinguish a copy from the original, if information is used to control access to resources, anyone who is able to copy the information will have access to those resources. Consequently, it is difficult to establish ownership of digital intellectual property: two parties may each claim to be the legitimate owner of digital information.

Digital watermarks are an attempt to address the problem of perfect copies in digital data. While they are not foolproof, they are a workable approach provided a few conditions are satisfied. Briefly, when using a digital watermark additional information is embedded into or superimposed on the original images. As concerns about establishing ownership of digital media have escalated in recent years (witness the claims by the recording industry blaming reduced sales of CDs on illicit file sharing), watermarks have attracted increased attention.

We may differentiate visible and invisible watermarks. Visible watermarks are often used in TV transmissions, where in a fixed location in each image or frame, a small logo identifying the transmitter is inserted, obliterating or obscuring that part of the image. Another type of visible watermark is provided by IBM's project watermarking a portion of the Vatican Libraries' holdings of images. Visible watermarks can typically be removed quite easily, thereby removing (a portion of) the information contained in the watermark. Of course, this creates the problem what to put in the place of the removed visible watermark if the watermark replaced it. Since it is not possible to restore the original image, an "empty" spot is left in the resulting image which could be filled by interpolation but this will frequently provide unsatisfactory results since it will still be possible to discern the (rough) shape of the removed logo, even though information contained in the logo would no longer be accessible. If the watermark was added instead, subtracting it restores the original image. For these reasons, invisible watermarks are preferred. Invisible watermarks change certain characteristics of the image, but this is done in a way that is not noticeable to the naked human eye. Here, we will consider exclusively invisible watermarks.

Depending on one's objectives, either robust or fragile watermarks can be used. Fragile watermarks have been proposed with the intention of degrading the watermark with each subsequent copy operation; thus, fragile watermarks are designed to limit the number of times a document may be passed on. Robust watermarks are of interest if one wishes to attach an indelible stamp of ownership; clearly the methods employed must be robust, that is impervious to various operations, such as rescaling, filtering, or superimposing an additional watermark. A variety of schemes designed to achieve these objectives have been proposed; see for example [Tanaka 1990, Matsui 1994, Bender 1995, Berghel 1997, Cox 1997, Barni 1998, Duan 1998, Lee 1999]. While robust and fragile watermarks can be considered complementary, it is the robust ones that serve in establishing ownership. In this paper, we will consider exclusively robust watermarks.

One aspect that has received little attention relates to the amount of information that can be encoded in a watermark. Clearly, robustness is directly correlated with the redundancy of the watermark; for example, if a certain small pattern is repeated many times in a watermark, the removal of the watermark through cropping an image is foiled. Similarly, the invisibility of a watermark is related to the extent of changes in the information that makes up the media; clearly,

extensive changes will have a greater impact on the watermarked medium than small ones. In general, it is useful to view the process of watermarking an image akin to injecting energy – the more energy is injected, the more the original image is impacted, to the point where this process may be perceptible to the viewer. This is of course the antithesis of invisibility! Thus, there are certain limits on the amount of information that can be encoded in the watermark. To alleviate problems created by the paucity of information available in the watermark, we propose the notion of time-variant watermarks. In this scheme, different frames of a video (or an audio) file will be associated with different watermarks. There are two advantages to this approach: it makes it much more difficult to defeat the watermark, and it lets one encode significantly more information in the watermark while permitting a great deal of redundancy and repetition. The most important practical aspect of this new scheme is that its computational complexity is identical to that of conventional watermarking: It is immaterial whether we embed the same watermark into many frames of the video or whether we embed a different watermark in each of these frames. It will become quite clear that the additional information in the watermark can be exploited to achieve increased protection of intellectual property.

All existing watermark schemes, visible and invisible, robust and fragile, are time-invariant: the embedded watermark is the same, independent of the video frame into which it is embedded. In contrast, our time-variant watermark scheme permits the embedding of sequences of watermarks into the medium to be watermarked. Typically, the watermark will consist of a number of frames that may be smaller than the number of frames of the medium into which the watermark is embedded. If it is smaller, then as in time-invariant watermark schemes, the watermark sequence is repeated until the end of the medium into which it is embedded is reached. This allows one either to increase the amount of information that is encoded in the aggregate watermark or to reduce the amount of information that is contained in a single watermark frame. Most interesting is the case where the number of watermark frames is equal to the number of frames of the file to be watermarked; here, the watermark can be used to insert sequencing information that is invisible to the viewer. This information can be used to detect, and demonstrate if required, whether original frames have been removed or permuted or repeated. In particular, the removal and the permutation of frames existing in a digital video cannot be detected using conventional, time-invariant watermarks: removing watermarked frames is cannot be detected, since everything remaining is properly watermarked, and the same argument applies to permuting existing frames. Note that inserting new frames is detectible using time-invariant watermarks since the new frames would not be watermarked.

We sketch our approach to time-variant, invisible, robust waterworks using video media; an analogous approach can be formulated for audio or similar media that represent information where a certain amount of errors can be tolerated. It is for example clear that viewers of television are perfectly willing to tolerate a fairly high percentage of "wrong" pixels, perhaps as high as 5% without major deterioration in the perceived quality of the image viewed. For audio the percentage may be somewhat lower, but still significant. On the other hand, an error rate of even 0.5% in text data would be considered quite unacceptable, as it amounts to about one error each three lines of text. This would be even more unacceptable if such "errors" were not randomly distributed, but instead deliberately introduced; here, even a single error (change of one character) might be devastating – consider for example a contract obliging A to pay US$10,000 and assume that the numeral "1" were changed to the numeral "9"!

An important aspect in storing and disseminating digital videos is the amount of data required to represent them faithfully. Clearly, we want to keep the file size as small as possible. Attempts to reduce the size of a video lead naturally to data compression techniques. Following industry standards, we assume JPEG encoding for individual (still) images as well as intracoded frames.

We assume a standard MPEG organization of the video sequence into I (intracoded), P (predictive-coded), and B (bi-directionally predictive-coded) frames. We outline our approach's advantages, in particular increased imperviousness against a variety of attempts to defeat the watermarking process, through filtering, cropping, resizing, and other operations, and quantify the increase in information content that can be accommodated in the new watermark.

We note that some of the objectives one pursues in using watermarks can be attained by other means, primarily encryption-based approaches [Leiss 1982]. For a discussion of these, see for example [Chen 1995, Chen 1996]. The current work is primarily based on research reported in two M. S. theses [Yang 1999, Yang 2001]. We will give a very brief review of goals and objectives in using watermarks. Then we give a sketch of the MPEG organization of a video file, with some attention paid to the representation of color and the JPEG technique for still images. Then we discuss time-variant watermarks in more detail and indicate the benefits obtained in this way. We conclude with a summary of the advantages of the approach and by indicating possible future work.

2. Watermark Goals and Objectives

We briefly review aspects of watermarks pertinent to our work [Bush 1999, Chun 1998, Cox 1997, Hsu 1998, Hsu 1999, Koch 1995]. The overall objective is the protection of intellectual property [Berghel 1997], in our case, the intellectual property contained in a digital video file.

As already mentioned, we are interested in invisible, robust watermarks. Robustness means that the watermark must be impervious to attempts at removing, destroying, obliterating, or overwriting it. Any attempt to do so should result in a very noticeable degradation of the image before the watermark is lost. Given the environment in which the watermark is used, the process of embedding the watermark must be compatible with MPEG processes. Consequently, watermarks must be able to survive both loss-less and lossy compression techniques, as well as other common video processing techniques, such as scaling, cropping, resizing, and filtering (in the case of color video, this includes changes in the color scheme, such as reducing the color palette [e. g., from 16 bit to 8 bit]).

The watermark must allow the legitimate owner of the video to demonstrate this ownership conclusively (for example, to a judge or adjudicator). Therefore, sufficient information be present that can be used for this purpose. Below we will argue that none of the existing, time-invariant watermark schemes fully attains this goal. The principal reason for this is the fact that within the context of MPEG-based compression, it is virtually impossible to guarantee that entire scenes have not been removed from the video nor that original scenes have been permuted or repeated. More specifically, a new scene in a video will almost certainly result in the use of an I-picture for the first frame of the scene. Since most of the watermark insertion concentrates on I-pictures, this implies that the removal of a group of frames (for example, an entire scene) that begins with an I-picture would not be noticeable if all watermark images are identical, that is, if they are time-invariant. It is true that the running time of a video could be used to **detect** this type of tampering, but it would not allow one to determine **where** the tampering occurred – for this, our time-variant watermark approach described in this paper is required.

Finally, we mention three important practical aspects of any watermark; failure to satisfy either one of them will render the approach unacceptable in practice:

1. The insertion of the watermark must not affect the perceived quality of the video. While the watermark information is of course embedded in the signal (i. e., the original signal is modified by the watermark insertion), this must not affect the **perceived** quality of the signal.

2. The process of inserting the watermark must not substantially increase the overall complexity of generating and using the video, at least not significantly beyond what MPEG already requires. This is the reason why certain cryptography-based signature schemes (see [Chen 1995, Chen 1996]) are not acceptable in practice, even though they could be made arbitrarily secure.

Clearly, if the quality of the video images is visibly affected by the watermark insertion, viewers will refuse to accept the resulting lower quality of the images. On the other hand, no matter how well the image quality is preserved, if the process of inserting the watermark adds a significant amount of processing to the already somewhat time-consuming MPEG processing requirements, there may simply not be sufficient compute power to carry out the watermark **insertion** in real time. Note that the complexity of **viewing** a watermarked video is not increased by the watermark, in contrast to encryption-based approaches.

3. It must be possible to demonstrate legally that only the true owner of a video is capable of embedding the watermark. If this is not possible in a legally binding way, the utility of a watermark for the protection of intellectual property is seriously compromised.

3. MPEG Compressing of Video Files

We sketch the organization of MPEG-compressed video files, starting with JPEG for still images which forms the basis of MPEG. First however we explain color representation, with its implications for the embedding of watermarks.

MPEG is essentially a (family of) method(s) for compressing a video file. An ordinary 24-bit image with 640*480 pixels requires almost 1MB of space (high-definition digital images would require even more). Since there are 30 frames per second in a typical digital video, a one-hour video amounts to about 100 GB of data. This amount of raw data contains a great deal of redundancy, the reduction of which is the goal of the use of (one of) the MPEG techniques. All MPEG schemes are based on the JPEG technique, applied to (some of the) individual frames of the video.

In virtually all videos, both individual frames themselves and the succession of frames contain much overt redundancy. For example, the background of a scene will ordinarily not change from one frame to the next unless the camera moves (temporal redundancy); moreover, this background may be virtually featureless and constitute a relatively large percentage of the frame (consider recording an interview), resulting in a large degree of redundancy (spatial redundancy).

Both the storage of the video file and the bandwidth requirements that result when transmitting digital video files over a network are a concern that data compression is designed to address. Were the above mentioned video file transmitted in its raw format, the minimum bandwidth necessary for sending a single digital video would be 240 Megabits per second (without leaving bandwidth for anything else). This is of course unmanageable (consider for example video-on-demand services).

JPEG [Wallace 1992] stands for Joint Photographic Experts Group, an ISO/CCITT committee and is a standardized compression technique for full-color or gray-scale images of realistic digital images. (It is not designed for line drawings or lettering although the presence of such features is not an impediment for JPEG.) It is based on a lossy compression technique known as the Baseline method; this is a scheme that employs the DCT (Discrete Cosine Transform, see [Wallace 1992]). A compression technique is called **loss-less** if the information content of the original file can be retrieved from the compressed file in its entirety, without sacrificing accuracy or precision. A technique is called **lossy** if the compressed file loses some of the original information. Although a loss-less approach appears more attractive, it is typically the lossy techniques that result in significantly larger savings. Most importantly, the loss of information they suffer is typically imperceptible to the viewer. It is not unusual to obtain a compression ratio (uncompressed file size compared with compressed file size) of 15 or more with excellent image quality; this compression ratio can be even higher if some deterioration of the image quality is acceptable [Sonka 1998]. JPEG, and consequently MPEG, allows the user to specify the image quality in terms of rather intuitive parameters. In this way, the image quality can be varied, depending on the given application. For example, a major motion picture may be encoded with greater faithfulness (and at greater cost in storage space or transmission bandwidth) than a video conference in a corporate setting. JPEG is considered a very popular and efficient coding scheme for continuous-tone still images. It also forms the basis of the MPEG family of approaches to encoding digital video. Before we describe MPEG-2 (which is at present the main representative of the MPEG schemes applicable to digital video), we give some brief explanation of digital color and its representation.

Humans perceive colors as combinations of the primary colors red, blue, and yellow (the typical rainbow arrangement). Only slightly deviating from this, video hardware generally uses the RGB model (Red, Green, Blue) with a pixel being associated with a triple (RGB) representing the color intensities; (000) represents black in this scheme (absence of everything), (kkk) white (presence of everything), (k00) pure red, and so on, where the value k is the quantization granularity for each primary color (for a total of k+1 different values, namely 0 through k). Thus, if k is 255 (a very common choice since it amounts to one byte), there are 2^{8+8+8} or 2^{24} different representable colors. Clearly, smaller values of k correspond to less faithfulness in the color scheme, larger values to greater faithfulness. With few exceptions (a contrary example might be a fairly uniform sky that continuously goes from light blue to gray), color schemes with more than 24 bits (eight for each of the three primaries) result in improvements in image quality that are virtually imperceptible to the unassisted human eye.

In practical applications, the RGB signal is usually transformed into one that is displayable with fewer major artifacts on black-and-white devices (including printers!), namely the (Y, C_b, C_r) representation, where Y is the luminance, C_b is the blue chrominance, and C_r the red chrominance. (R, G, B) and (Y, C_b, C_r) correspond to each other linearly [Benoit 1997]:

$$Y = 0.587\ G + 0.299R + 0.114\ B$$
$$C_b = 0.564\ (B - Y)$$
$$C_r = 0.713\ (R - Y)$$

It is important for the design of data compression techniques to understand that the human eye is less perceptive for color than for luminance. This implies for natural images that the chrominance components of a signal can tolerate a more reduced bandwidth than the luminance component, without affecting significantly the perceived image quality. Typically, the bandwidth for chrominance may be chosen to be one half to one quarter of that for luminance without affecting human perception [Benoit 1997].

Lossy JPEG compression consists of six main steps [Wallace 1992]:

1. Decomposition of the image into blocks of size 8*8 pixels; each block can be viewed as a 64-point discrete signal which is a function of the two spatial dimensions.

2. The Discrete Cosine Transform is applied to each 8*8 matrix which generates a new 8*8 matrix consisting of the coefficients of increasing spatial frequency. These coefficients can be viewed as the relative amount of the 2D spatial frequencies in the 64-point input signal. The coefficient with frequency 0 in both dimensions is referred to as the DC coefficient while the other 63 are the AC coefficients.

3. Quantization (or discretization) is applied to the 64 DCT coefficients to yield an 8*8 Quantization table Q(u,v) consisting of integers. As result of the DCT operation, the values in Q increase from left to right and from top to bottom. This takes into account the peculiarities of human vision, in particular the fact that the human eye does not distinguish very fine details below a certain luminance level.

4. The 63 AC coefficients in Q are concatenated into a zigzag scan; in terms of (u,v), this scan is

DC: 00

AC: 01 10
20 11 02
03 12 21 30
40 31 22 13 04
05 14 23 32 41 50
60 51 42 33 24 15 06
07 16 25 34 43 52 61 70
71 62 53 44 35 26 17
27 36 45 54 63 72
73 64 55 46 37
47 56 65 74
75 66 57
67 76
77

This helps in entropy coding by placing low-frequency coefficients, which are more important in perception, before high-frequency ones.

5. Run-length coding replaces a sequence of identical values by one indication of that value followed by the number of these values in the sequence. This is where major compression in JPEG occurs, since from a certain point p on in the sequence of the 63 AC coefficients of the zigzag scan, we can replace the remainder by zeroes without affecting the visual quality of the image. The value of p is a parameter in this process: if p is small, say 5, the image quality is reduced and the compression greatly improved; if p is large, say 30, the image quality is virtually unaffected but at the cost of reduced compression. In fact, studies of human perception of typical images have shown that even for relatively small values of p, say around 10, the perceived quality of the image is virtually unaffected. While the value at which people will notice a difference depends on the type of image, it is a very important aspect of JPEG to determine as small a value of p as is acceptable from a visual perception point of view.

6. The final step consists of applying Huffman coding to the resulting sequences; this further reduces the amount of data to be transmitted.

MPEG is based on JPEG and is designed to remove temporal redundancies (redundancies that occur from one frame to the next) after JPEG has been applied to remove the spatial redundancies within each frame. Temporal redundancies are detected by motion estimation whereby portions of images in consecutive frames are matched up. Three fundamental types of pictures are distinguished in this process, namely I-pictures, P-pictures, and B-pictures. Intra or I-pictures are encoded without any reference to other frames, while Predicted or P-pictures and Bi-directionally Predicted or B-pictures depend on other frames, for P-picture only on the preceding I- or P-picture, for B-picture on I- and P-pictures both preceding and following it. The number of P-pictures between two consecutive I-pictures is an important parameter: Since much redundancy is detected (and removed!) between I-pictures, making this value large results in more savings. However, making it too large will affect the quality of the interpolated image frames. B-pictures fill in the gaps between I- (and P-) pictures and provide the largest savings. The objective is to have as few I- (and P-) pictures as is possible without affecting the visual quality of the video. Since typically there are many more B-pictures than I- or P-pictures, ratios of 200 can be achieved in video compression without sacrificing a great deal of quality [Sonka 1998]. This value of 200 is the combination (product) of the JPEG compression ratio and compression ratio resulting from the removal of redundancy related to motion estimation, based on the I-, P-, and B-pictures. To give a few specific values, if M (N) is the number of pictures between two successive P-pictures (I-pictures), then typical values might (3,12). Thus, 1/12 of a group of pictures are I-pictures, 1/4 P-pictures, and 2/3 B-pictures.

Motion estimation involves defining a motion vector, which establishes the correlation between a "departure" zone in the first picture and an "arrival" zone in the second. This is done on the basis of macro blocks (blocks of size 16*16, or four 8*8 blocks of luminance, one 8*8 block for red chrominance, and one 8*8 block for blue chrominance). This allocation of four times the amount of data for luminance than for each of the chrominance values reflects the differing levels of perception of the naked human eye.

4. Time-Variant Watermarks

First, we briefly review the process of embedding a (time-invariant) watermark into an MPEG-2 digital video file. In essence, our approach is applicable to any watermarking scheme. Thus, we are less interested in a specific scheme; instead, we describe the differences between the traditional, time-invariant approach and our time-variant method, outline the advantages of our approach, and indicate how it can be used to attain higher levels of protection of intellectual property.

Numerous approaches to embedding (time-invariant) watermarks into images have been described in the literature. They can be grouped into two major categories, namely methods that embed the watermark by modifying directly the intensity of (some or all of) the pixels of an image [Bender 1995, Nikolaidis 1998], and methods that act upon (some or all of) the coefficients of an underlying transform domain (most common are the Discrete Cosine Transform or DCT and the Discrete Fourier Transform or DFT) [Koch 1994, Boland 1995, Cox 1997, Barni 1998, Duan 1998, Lee 1999]. While we concentrated in [Yang 1999, Yang 2001] on the method described in [Cox 1997], it should be clear that any of the domain-based approaches would do nicely in an MPEG environment. The underlying idea is the following notion known as spread spectrum technique: The frequency domain of the image to be watermarked is viewed as a communication channel and the watermark is viewed as a signal that is transmitted through it.

Thus, the watermark is spread over many frequencies so that the energy change in any one frequency is small enough to render it imperceptible. The objective is of course that the embedded watermark survive common signal manipulations (such as lossy and loss-less compression, filtering, conversions between digital and analog representation) and geometric manipulations (such as cropping, scaling, translation, rotation). In addition to these, superposition of one or more additional watermarks should also be detectable. Finally, manipulations related to sequencing of pictures in a video are of concern; these include in particular adding new or removing original pictures. Another requirement relates to the ability to demonstrate conclusively to a judge one's ownership of the original video, that is, the owner, and only the owner, should be able to do this. We refer to the literature for the technical details of inserting the (time-invariant) watermark. For our purposes, it suffices to note that techniques exist which meet the stated requirements and which are sufficiently simple and efficient to permit their implementation within the context of an MPEG-2 video file without increasing the complexity of the operations involved in generating, viewing (or possibly removing the watermark), or adjudicating a watermarked video [Busch 1999].

An important aspect of watermarking within an MPEG context is the determination which pictures of a video file are to be watermarked. On the basis of our brief description, it is clear that the watermarking process involves individual frames or pictures which are subjected to JPEG compression. This implies that the watermark should be inserted into the AC coefficients that occur quite early in the zigzag scan, since later AC coefficients may simply be removed (set to 0) without affecting the visual quality of the image. There are different techniques that ensure that the injection of energy (that is, the embedding of the watermark) into these coefficients does not distort their values unduly. As noted in [Benoit 1997, Katzenbeisser 2000], this approach is robust and affects the visual quality only minimally. Given the process of MPEG compression, we have three types of pictures, I-, P-, and B-pictures. Since only I-pictures are independently encoded in MPEG, watermark insertion concentrates on I-pictures. However, this does not imply that P- or B-pictures are unaffected, in as much as they depend on (watermarked) I-pictures (as they are interpolated based on these pictures) and thus are indirectly watermarked.

A watermark is typically an image that is substantially smaller that the video frame; for example, assuming a 640*480 pixel image, the watermark may be of size 80*60. Also, it may be black-and-white in order to reduce the amount of information that must be accommodated in each video picture. The watermark image would then be repeated 64 times to fill the entire image region.

Time-invariant watermarking schemes embed the same watermark picture into all pictures that are explicitly watermarked (essentially all the I-pictures). In contrast, our approach to providing time-variant marking schemes takes a watermark **video** consisting of a number N_0 of pictures and embeds this video in the usual way, frame by frame. Specifically, into the video frame number i to be watermarked we embed the watermark frame number i, for i=1, 2, …, N_0. If there are more than N_0 pictures in the video to be watermarked, the next batch of N_0 pictures get a second copy of the watermark video embedded, and so on, until the end of the video to be watermarked is reached. The number N_0 is a parameter: if N_0=1, then we have the ordinary, time-invariant watermarking scheme; if N_0 is greater than 1, the approach is time-variant. A sensible upper bound for N_0 is the number of I-pictures in the original video.

The information contained in the N_0 frames of the watermark is entirely up to the user. It is however useful to provide some sequencing information in the watermark video because if N_0 is equal to the total number of I-pictures in the video this will enable one to ensure that no original pictures had been removed from the watermarked video. Note that this is one operation that

traditional, time-invariant methods are entirely unable to detect since the removal of an entire scene (starting with an I-picture) is undetectable. Other subversions that time-invariant watermarks are unable to detect, but that time-variant watermarking handles with ease, are permutations and repetitions of existing (that is, properly watermarked!) frames in the video.

We remark that in both time-invariant and time-variant watermarking, information is injected into the signal corresponding to each of the watermarked frames, the time-invariance of this information is very wasteful. In contrast, although our approach will inject no more energy into each of the watermarked images than the traditional methods, the information content our approach allows us to embed is dramatically greater, since it changes from one watermark frame to the next. Furthermore, the time-complexity of inserting a time-invariant watermark is identical to that of inserting a time-variant watermark. Thus, time variance provides significantly greater functionality at no cost whatsoever!

The following table summarizes in what way an attack against a watermark is foiled; here INV indicates that the traditional time-invariant watermarking scheme will guard against this attack or manipulation (preserving the watermark) while VAR indicates that this is achieved by time-variant watermarks:

loss-less compression	INV, VAR
lossy compression	INV, VAR
filtering	INV, VAR
conversion (digital ↔ analog)	INV, VAR
cropping	INV, VAR
scaling	INV, VAR
translation	INV, VAR
rotation	INV, VAR
superposition of another watermark	INV, VAR
adding new frames	INV, VAR
removing original frames:	**VAR**
permuting original scenes:	**VAR**
repeating original scenes or frames:	**VAR**
legal demonstration of ownership:	INV, VAR

Recently, an experimental implementation of time-variant watermarks has been carried out by Ms. Enohi Ibekwe as part of a graduate project in Information Assurance. Specifically, she incorporated Koch and Zhao's method [Koch 1995] into MPEG-1 encoded video. This was done using the Dali multimedia library on a scripted language (TCL). Watermarks were inserted only into I-frames (which are compressed independently of any other frames), while the P- and B-frames were not watermarked; note however that vestiges of watermarks are present in these frames as well, since they depend on the (watermarked) I-frames. The results confirmed the observations above.

5. Conclusions

We have outlined our approach to embedding time-variant watermarks into digital video files. This method permits a significant increase of the amount of information over conventional, time-invariant watermarks while retaining all the advantages of conventional watermarking. In particular, the additional cost incurred by introducing time-variance is zero. The approach was

formulated assuming the video file is represented in an MPEG-2 format, involving I-pictures, P-pictures, and B-pictures. In view of the standard data compression algorithm underlying MPEG-2, frames of the watermark video are embedded into the I-pictures, that is, those pictures of the video that are encoded independently, using JPEG techniques. We discussed implementation issues of time-variant watermarks, as well as their advantages over the usual time-invariant watermarks. In particular, this watermarking scheme permits one to defeat not just the usual attacks involving filtering, cropping, resizing, and changing color schemes, but also to guard against new attacks, namely removing or repeating frames as well as permuting scenes of the video. Important is that the complexity of the operations of embedding the watermark, viewing the watermarked video, removing the watermark from the video, and the adjudication of the watermark remains unaffected by the watermark. Specifically, embedding of the watermark is incorporated in the MPEG-2 compression scheme and adds an insignificant amount of work; viewing the video is completely unaffected by the watermark, removing the watermark amount to MPEG-2 compression, and adjudication is essentially equivalent to extracting the watermark (which is in turn the same as removing it).

Bibliography

[Barni 1998] M. Barni, F. Bartolini, V. Cappellini, and A. Piva: A DCT-Domain System for Robust Image Watermarking, Signal Processing 66, 1998, 357-372.

[Bender 1995] W. Bender, D. Gruhl, and N. Morimoto: Techniques for Data Hiding, Proc. SPIE 2420, 1995, 164-173.

[Benoit 1997] H. Benoit: *Digital Television MPEG-1, MPEG-2 and Principles of the DVB System*, Arnold London, UK, 1997.

[Berghel 1997] H. Berghel and L. O'Gorman: Digital Watermark, http://www.acm.org/~hlb/publications/dig_wtr/dig_watr.html, 1997.

[Boland 1995] F. M. Boland, J. J. K. O Ruanaidh, and C. Dautzenberg: Watermarking Digital Images for Copyright Protection, Image Processing and Its Applications, 1995, 326-330.

[Bush 1999] C. Bush, W. Funk, and S. Wolthusen: Digital Watermarking Using DCT Domain Constraints, Proc. Int'l Conf. Image Processing, Vol. 3, 1996, 231-234.

[Chen 1995] F. Chen: Multimedia Authentication, M. S. Thesis, December 1995, Department of Computer Science, University of Houston.

[Chen 1996] F. Chen and E. L. Leiss: Authentication for Multimedia Documents, Proceedings, CLEI PANEL'96 - Conferencia Latinoamérica de Informática, June 3-7, 1996, Bogotá, Colombia, 613-624.

[Chun 1998] T. Chun, M. Hong, Y. Oh, D. Shin, and S. Park: Digital Watermarking for Copyright Protection of MPEG2 Compressed Video, IEEE Trans. Consumer Electronics, Vol. 44, No. 3, 1998.

[Cox 1997] I. Cox, J. Killian, T. Leighton, and T. Shamoon: Secure Spread Spectrum Watermarking for Multimedia, IEEE Trans. Image Processing, Vol. 6, No. 12, 1997, 1673-1687.

[Duan 1998] F. Y. Duan, I. King, L. W. Chan, and L. Xu: Intra-Block Algorithm for Digital Watermarking, IEEE Proc. Int'l Conf. Image Processing, Vol. 2, 1998, 1589-1591.

[Hsu 1998] C. Hsu and J. Wu: DCT-Based Watermark for Video, IEEE Trans. Consumer Electronics, Vol. 44, 1998, 206-216.

[Hsu 1999] C. Hsu and J. Wu: Hidden Digital Watermarks in Images, IEEE Trans. Image Processing, Vol. 8, No. 1, 1999, 58-68.

[Katzenbeisser 2000] S. Katzenbeisser and F. A. P. Petitcolas: Information Hiding Techniques for Steganography and Digital Watermarking, Artech House, Boston, MA, 2000.

[Koch 1994] E. Koch, J. Rindfrey, and J. Zhao: Copyright Protection for Multimedia Data, IEEE Proc. Int'l Conf. Digital Media and Electronic Publishing, 1994, 203-213.

[Koch 1995] E. Koch and Z. Zhao: Toward Robust and Hidden Image Copyright Labeling, Proc. IEEE Workshop Nonlinear Signal and Image Processing, 1995, 443-449.

[Lee 1999] C.-H. Lee, H.-S. Oh, Y. Baek, and H.-K. Lee: Adaptive Digital Image Watermarking Using Variable Size of Blocks in Frequency Domain, Proc. IEEE Region 10 Conf. on TENCON, Vol. 1, 1999, 702-705.

[Leiss 1982] E. L. Leiss: *Principles of Data Security*, Plenum Publishing Corp., New York, NY, 1982.

[Matsui 1994] K. Matsui and K. Tanaka: Video Steganography, J. Interactive Multimedia Association Intellectual Property Project, Vol. 1, No. 1, 1994, 187-206.

[Nikolaidis 1998] N. Nikolaidis and I. Pitas: Robust Image Watermarking in the Spatial Domain, Signal Processing, Vol. 66, No. 3, 1998, 385-403.

[Sonka 1998] M. Sonka, V. Hlavac, and R. Boyle: *Image Processing, Analysis, and Machine Vision*, PWS, Pacific Grove, CA, 1999.

[Tanaka 1990] K. Tanaka, Y. Nakamura, and K. Matsui: Embedding Secret Information into a Dithered Multilevel Image, Proc. 1990 IEEE Military Communications Conf., 1990, 216-220.

[Wallace 1992] G. K. Wallace: The JPEG Still Picture Compressing Standard, IEEE Trans. Consumer Electronics, Vol. 38, No. 1, 1992, 18-35.

[Yang 1999] Z. Yang: Time-Variant Watermarks in Digital Video, M. S. Thesis, December 1999, Department of Computer Science, University of Houston.

[Yang 2001] Q. Yang: Time-Variant Watermarks in Color Digital Video, M. S. Thesis, December 2001, Department of Computer Science, University of Houston.

Personal Information Retrieval Visualization (PIRV): Clustering and Visualization of Web Document Search Results

Xiangyang Xu and Ernst L. Leiss
Department of Computer Science
University of Houston
Houston, TX 77207
{x228917@yahoo.com, coscel@cs.uh.edu}

Abstract

Conventional web search engines often return long lists of ranked documents as their output. This text-like data presentation for web search results has many limitations. Since only a part of the list of documents can be shown at a time, users cannot get a complete picture of the returned documents. Therefore, users do not know if these documents contain a document they are interested in, after reading the first few items of the list of documents. Due to the imprecise nature of current Web search engines and the explosive increase in the number of documents available, users are forced to spend a significant amount of time going through the list of the results or abandon the current search result.

In this project, we design and implement a system called PIRV (Personal Information Retrieval Visualization), which dynamically groups the search results into clusters and presents these clusters in 2-dimensional graphics. After receiving a query from a user, PIRV sends it to the search engine, receives the returned documents, clusters these documents according to similarity values between individual documents, transforms the data into a graphical representation, and then displays these graphics to the user. With this visual display, a user may use visual perception to evaluation these clusters and to make an intuitive judgment about the relevance of these documents without having to read a significant portion of each document. Furthermore, a user's search history is saved in the user's computer upon logging out; this can be used to assist in future searches. The saved search history file is automatically retrieved by PIRV upon login. A user can also view previous search results when doing multiple query searches.

Keywords: Internet search, clustering of results, visualization.

1. Introduction

Although much diverse information is available when using major search engines, users often suffer from too many results ([1, 2, 3]). Search engines often return excessively many documents or Web pages [4]. These results are then presented as an ordered list of documents with title and/or a brief text summary or description. Because the sheer size of information in the Internet and the imprecision of Web search engines, the number of returned documents through search engines can be huge. Furthermore, only a small part of a long ordered list of documents can be shown at a time.

Filtering the long ranked lists of returned document manually by the user may be time-consuming [1]. Since most of the documents are likely irrelevant, users must filter out many irrelevant documents before finding out what they really want. Users have to spend time going through long lists of documents, ignoring most documents after looking over the first entries in a list, or make additional queries, or switch to another search engine. This process is time-consuming and cumbersome; Web users may get frustrated with this information retrieval when the right web document cannot be found easily.

The records of users' query search histories are not maintained in most Web search services [5]. Users are likely to make multiple queries in order to find out what they want. Multiple queries may give users a set of returned documents based on refined queries or different queries. Users can get some ideas from the documents returned from multiple queries. Often a user may reformulate a query based on the results of multiple queries to improve the quality of search. However, most research engines do not provide the capability to store the query search history. Users may get lost or confused when the multiple query approach is used.

We designed and implemented an interface called "Personal Information Retrieval Visualization" (PIRV). PIRV was designed for clustering the retrieval results returned from a popular search engine (Yahoo [6]) and displaying the clusters visually to the user. Both graphical display and textual display are used in order to give users more information. The information visualization interface is our main focus, with textual display as a supplement. The

Web search results are clustered into groups through a clustering engine using the agglomerative hierarchical clustering algorithm. Users can perceive the clusters straightforwardly. PIRV also provides capabilities for users to manipulate the clusters dynamically. By examining the clusters, users save time and effort since they need not examine the data item by item. Additionally, facilities for storing previous queries are provided.

2. **Web Document Clustering**

In the text of each document extracted from HTML pages, there are many symbols such as numbers, punctuations, and signs, as well as short-words with fewer than four characters, such as a, the, of, is, are, etc. These non-word symbols and short-words are not used in the similarity calculation; they are removed from the string of text representing each document. Only the remaining words are kept and saved.

We group web retrieval documents into clusters according to key word relevance ([7], [8]). It is assumed that documents with the same topic are similar in their descriptions; these will be put into one cluster. Thus, the returned documents by search engines can be clustered into different groups according to their similarities. Our method to cluster the documents is to compare each key word in one document to the key words in every other document. The more matched key words there are between the two documents, the more similar the two documents are considered. The value of similarity between two documents is calculated according to the frequency of matched key words.

Measurement of content similarity between two documents is defined as the score of similarity. We have chosen to use Sorenson's similarity [9]. Sorensen's similarity between two documents X and Y is calculated according to the formula:

$$s = 2a / (2a + b + c)$$

where s = Sorensen's similarity
a = number of common key words in X and Y
b = total number of key words in X - a
c = total number of key words in Y – a.

Sorensen's similarity considers the balance between the total number of key words and the number of common keywords. It also considers the situations of too few common key words. The more common key words there are, the greater Sorensen's similarity is. If two documents have identical keywords, the Sorensen similarity is 1. For documents X and Y, there are two Sorensen similarity values, i.e., X compared to Y and Y compared to X. Since these two values are the same, all Sorensen's similarity values are saved in a triangular table for later clustering usage.

Hierarchical clustering groups documents into a hierarchical tree of clusters [10]. Agglomerative hierarchical clustering (AHC) [10] is the most popular type of clustering procedure and is commonly used for document clustering. Studies indicate that AHC produces clusters with high quality [11]. In this project, we used the AHC algorithm to group the documents into clusters according to their Sorensen similarity triangular table or Sorensen matrix [10]. Initially, every document is regarded as a cluster. There are three steps to find a cluster. First of all, we search the table and find the two clusters that have the closest similarity values. Then, these two clusters are combined into a new one. The average of the two similarities is the new combined similarity score. The third step is to recalculate the triangle table, i.e., the similarities between this new cluster and others are recalculated. We repeat this process until a given number of clusters is left. A user can select this number of clusters when making a query search.

In this project, we improved the AHC algorithm to reduce its time complexity. Previously, the computing time for the AHC algorithm had been reported to be $O(n^4)$ [12]. In our project, the similarity values are built into a triangular table. If there are n documents initially, the initial matrix requires n(n-1) / 2 calculations. However, after two documents are merged into a cluster, recalculating the triangle matrix requires on average $O(n)$ time (for one merge) provided the data structure for the matrix is properly organized. If these documents are organized in a Max heap structure, finding of the document with the maximum value can be done in time $O(1)$. Thus, the overall complexity is as following:

$$\frac{n(n+1)}{2} + \sum_{i=k}^{n-1} O(i) = O(n^2)$$

3. System Design and Architecture

The PIRV system architecture is based on a client-server model as shown in Figure 1. The system can be divided into three parts: 1) the far end components that include the query search subsystem, query search history subsystem, and user account subsystem, 2) the communication component, here the Apache Tomcat 4.0.4 server, that transports data between the client and the server, and 3) the client-side display components that display the results and allow users to interact with the system.

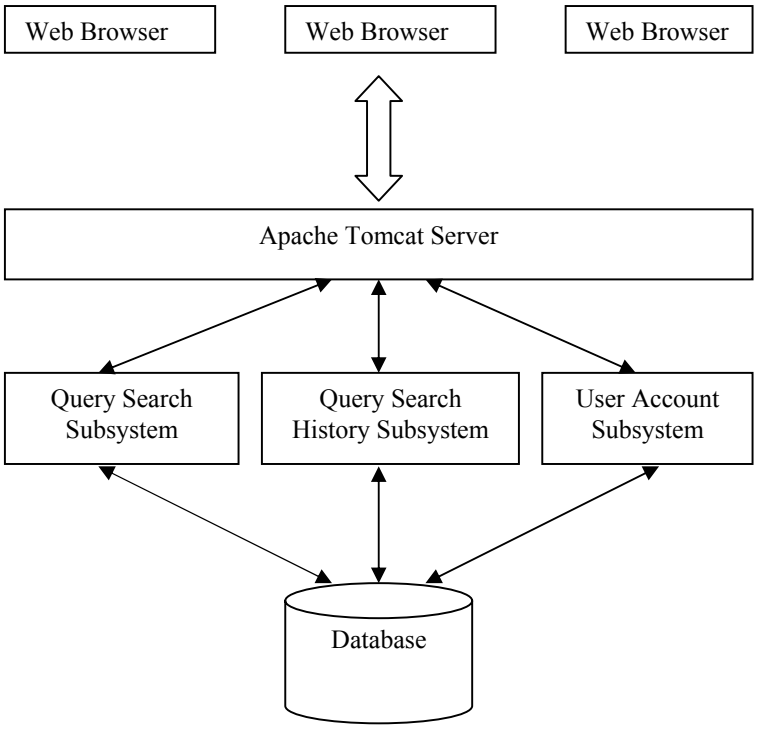


Figure 1 System Architecture of Personal Information Retrieval Visualization (PIRV)

In this project, an interface called "Personal Information Retrieval Visualization" (PIRV) was designed and implemented. PIRV was designed for clustering the retrieval results returned from a popular search engine (Yahoo [6]) and displaying the clusters visually to the user. It is a search engine using JSP and Applets for the GUI layer and using Java for retrieving and document clustering for the Apache Servlet ([13], [14]).

The query search subsystem is the major part of the entire PIRV system. It receives a user-entered query, sends the query to the search engine, collects documents from the search engine, produces clusters from these documents, and displays a graphical presentation of the cluster to the client. We have implemented the query search subsystem in four modules: 1) document retrieval module; 2) result processing module; 3) document cluster module; 4) data presentation interface.

The document retrieval module of the query search subsystem is initiated when a user submits a query via a browser. After receiving the user-entered query, the Java Servlet will transform the query into the URL-encoded format and send the query to the underlying search engine (Yahoo [6]). We use the Java URLconnection class to connect to the search engine. Then, the returned HTML page from the search engine will be received through the same connection channel. The returned HTML pages will be parsed.

The target of the result-processing module is the text of each document retrieved. Non-character symbols and other information must be removed. There are many "stop" words such as a, is, the, are, to, at, on, etc., which should be removed from the text since these "stop" words contribute little to the content of the description of each document. The text is first converted into lower case.

In order to make the document clustering more meaningful to users, the PIRV system gives the users the capability to specify how many clusters to create and how many results to retrieve from the search engine for this query. When a user makes a query, he can choose a number of clusters, from 3 to 15. The user can choose the number of documents, from 10 to 5000 from a Jlist, or all of the results from the search engine. He can also enter a number to specify the number of results.

The visualization of clusters is implemented on a class that extends the canvas. One of three buttons such as "Bar", "Pie", and "Dot" gives the signal and initializes the redraw process on the same canvas. Instances of three inner classes, chartCluster, pieCluster, and dotCluster, are used to represent the individual components such as bars in the bar graph view, pie slices in the

pie view, and circles in the dot view. These components will draw themselves on the canvas according to the values of the parameters that they have received. The components defined by the instances of these three inner classes all have a function to report their identities when users click on them so that the applet has to show the cluster information in the display panel.

The PIRV system has a query search history subsystem in order to allow users to see the results of previous queries. The search history may be important to some users who will do multiple queries and refined queries. Current query search results are temporarily saved in a server-side database so that the user can view them. When the user logs out, the query search results are sent to and saved in the user's computer upon request. The data in the server-side database are deleted. Next time when the user logs in, the file stored in his computer is automatically read and sent to the PIRV system. The data are extracted and installed in the server-side database.

The database was developed to hold account information and query search data of users. The tables are designed to hold query search information such as queries, clusters, and documents. Considering that writing to or reading from any text file is time-consuming, no text file was used in this project. Using the driver of the JDBC/ODBC Bridge is one of the common methods for a Java Servlet to support database connectivity [13]. We need both the ODBC driver and the JDBC/ODBC Bridge to establish a connection to the relational database. To access the database efficiently, we have built a database connection class, which has a constructor to make a connection to the database and has many functions to manipulate the contents of tables.

4. Paradigms in Visualization of Web Document Search Results

Information visualization investigates methods to support the exploration of large volumes of abstract data using graphical representations so that users may use their visual perception to evaluate and analyze the data ([15], [16], [17], [18]). This involves ways to transform the data into graphical expression. The data presentation interface in this project is to represent clusters in vivid 2-d graphic formats so that users may use their visual perception to evaluate and analyze the query search results. We have created three types of data presentation interface to

visualize the clusters, namely the bar graph view, the pie view, and the dot view. The three vitalization interfaces are implemented in separated canvases, which sit on three panels inside the Java applet. There are three command buttons for each panel. If a query search is successful, the three view buttons become active. The user can choose which view he likes by clicking on the corresponding command button.

The bar graph view arranges the clusters in the form of charts (see Figure 2), which is located on the left side of the applet. Each bar represents a cluster. The height of each bar represents the amount of documents of this cluster. The bars have different colors allowing users to differentiate the bars easily. The user clicks a bar to select the corresponding cluster. The scrolled list of documents of this selected cluster will show up in the display panel on the right side of the applet. Each document displays its document sequence number, a hyperlinked title, and a description. If the user wants to go to the page he is interested in, he clicks on the title and the selected page will show up in a separate window.

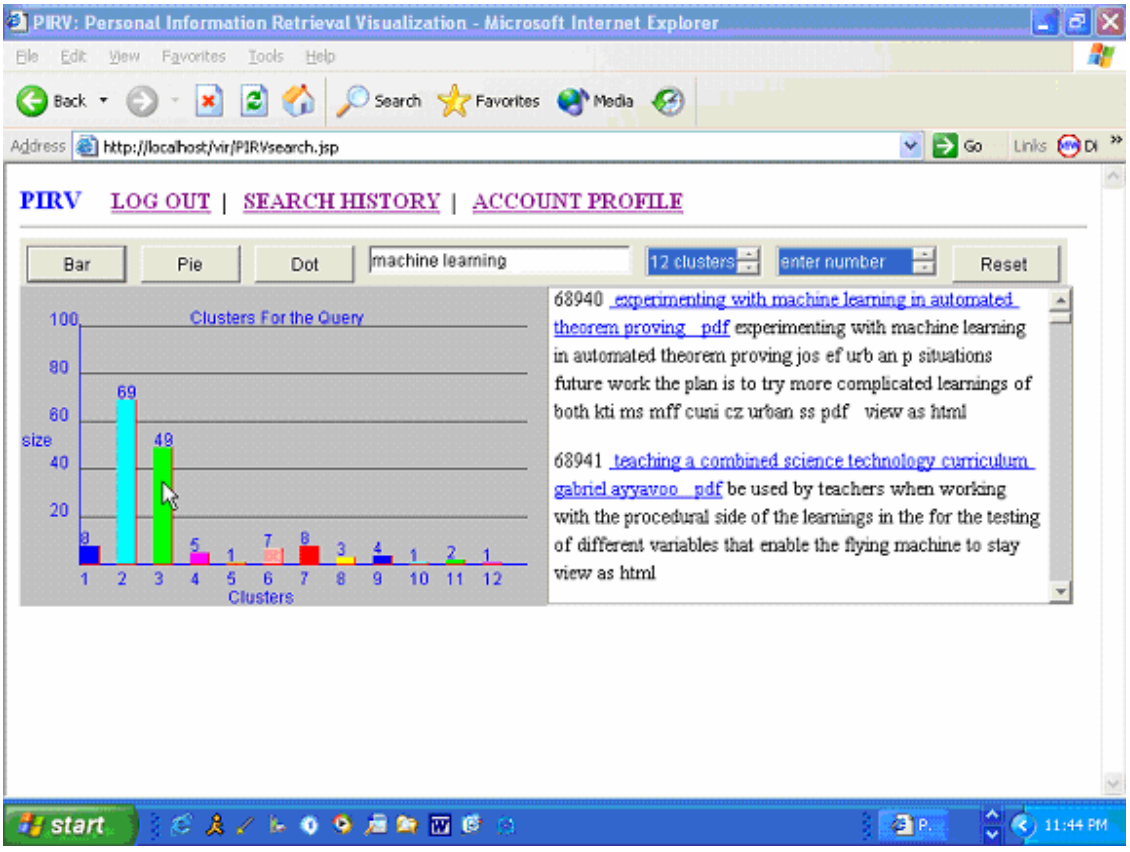


Figure 2 The Bar Graph View of the Data Presentation Interface

The pie view is to arrange the clusters in the form of a pie graph (see Figure 3). If the user clicks the pie button on the top of the applet, the pie view panel appears in the same place as the bar graph view. The pie is divided into different slices. Each cluster is represented by a pie slice that has a different color from its neighbors. The relative size of each piece in the pie represents the number of documents in this cluster. Similar to the bar graph view, the slices of the pie have links to their corresponding scrollable document lists that are shown in the display panel if selected. The user may easily select and view an individual cluster by clicking the corresponding slice in the pie graph.

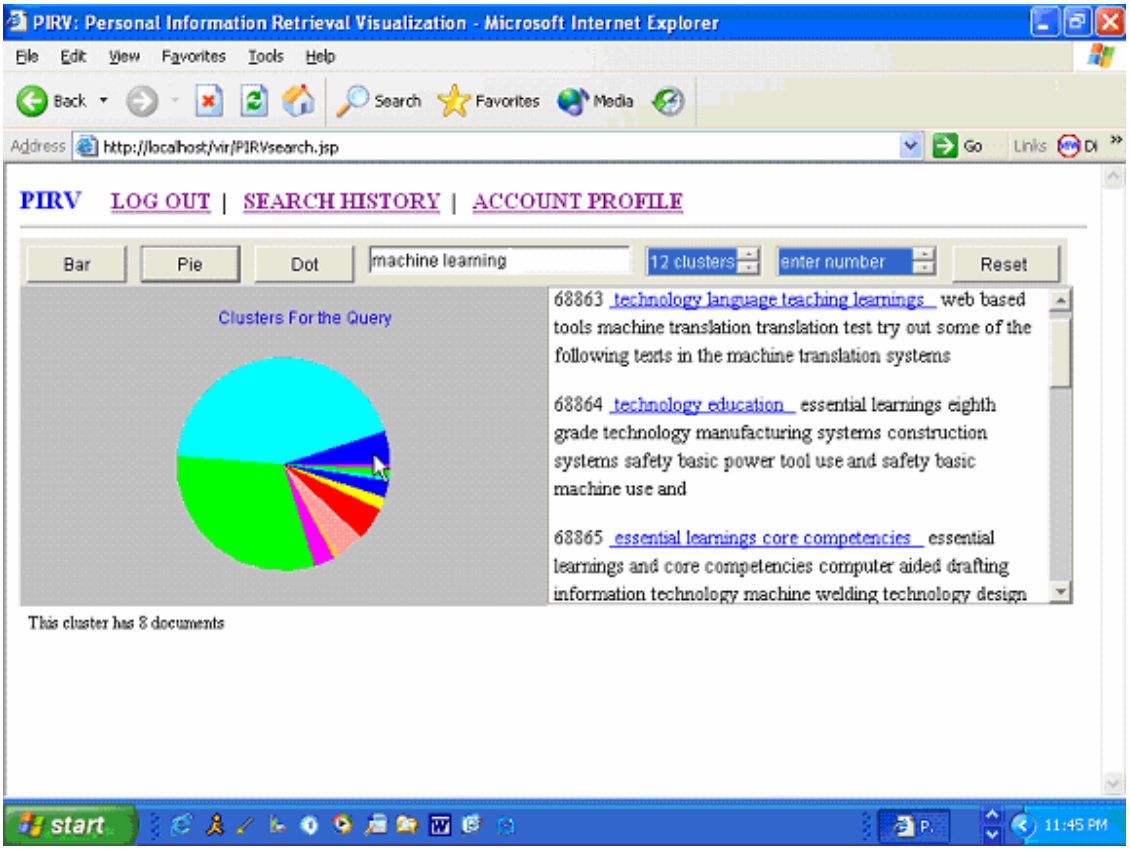


Figure 3 The Pie View of the Data Presentation Interface

The dot view is a concentric circular graphic with the dots located on it (see Figure 4). The graph has three concentric circles that give users a distance perception to the center. Dots represent the clusters. These dots have differing sizes, colors, and distances from the center.

The size of a dot represents the number of documents in this cluster. The colors of the dot allow the user to differentiate the clusters easily. The distance from the center is dependent upon the relevance to the query. The closer to the center, the more relevance to the query the dot has. Also, each dot is linked to the cluster information that is shown on the right side of the screen. The user may click a dot to select the corresponding individual cluster.

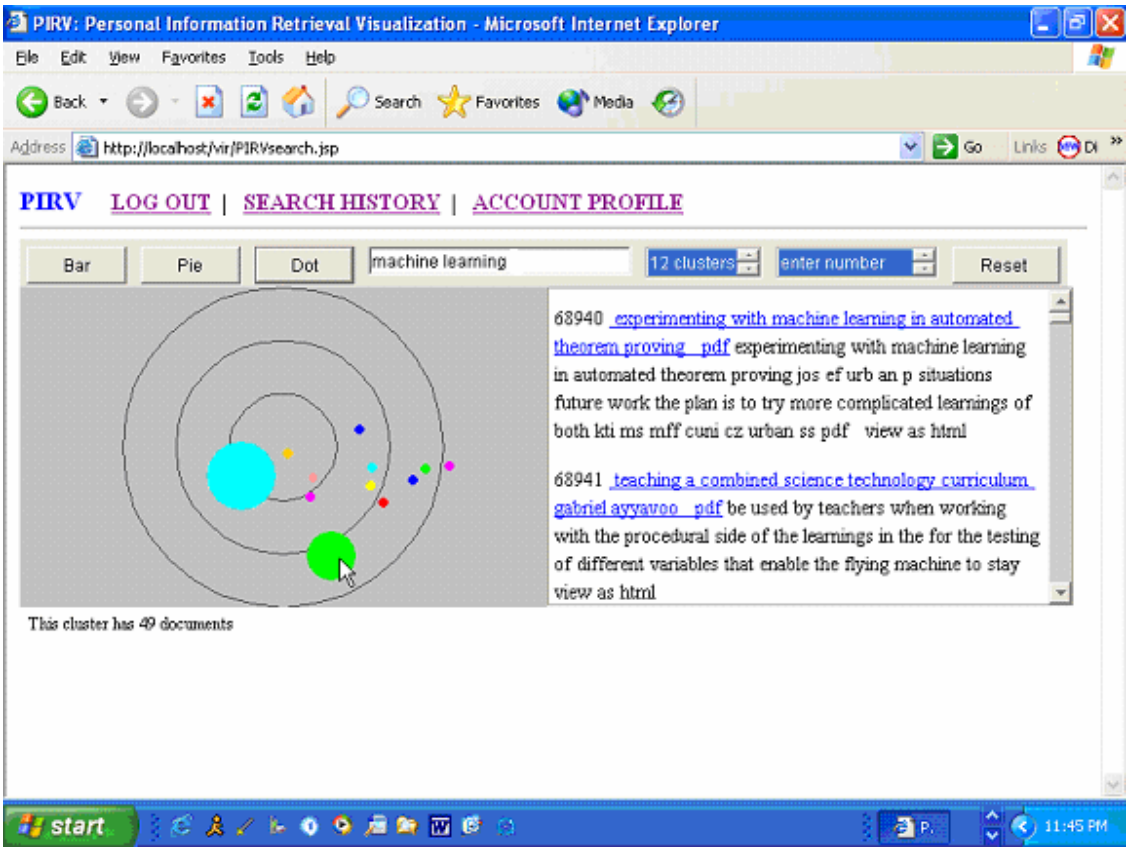


Figure 4 The Dot View of the Data Presentation Interface

5. Conclusions

In this paper we presented a Personal Information Retrieval Visualization (PIRV): a query search service tool with the capabilities of clustering and visualizing Web document search results. It provides the following features:

1) The system is an online system so that users can login into the system from anywhere in the world using a web browser.

2) Users can utilize this software to do query searches. The retrieval results are clustered and the clusters are displayed in the front-end so that users can view and browse these results.
3) The software provides three visualization panels that are serviced by the same data manager module. Users can visualize the retrieval results in three different ways, namely the bar graph view, the pie view, and the dot view.
4) The software provides a search history function. Users can save results between sessions. Subsequently, they can view and manipulate their saved search results.
5) The software design is independent of the search engine. PIRV can be used based on any search engine or meta-search engine with minimal coding changes.

The implementation of this system should have benefits for users by alleviating the difficulty of displaying many search results from Web search engines. Because our visual interface also provides the clusters' relevancy to a query, it allows users to make an intuitive judgment about the relevancy of documents. This tool may allow users to weed out quickly irrelevant clusters and concentrate on one or more relevant clusters. In this way, PIRV should be particularly useful when a large amount of results are retrieved from Web search engines.

We have identified a number of areas in which the design and implementation of the system can be enhanced. A study of evaluation of the PIRV display is needed to answer the question how much the clustering and visualization of this tool help users to find relevant documents more efficiently. More information about each cluster may be provided in order to give user a general idea about this cluster. This may need more computation in the server. The visual interface itself can be further improved by allowing users to zoom in on the clusters. In this way, a user can concentrate on a subset of documents and see more graphical detail.

Bibliography

[1] Eun-II Cho and Sung Hyon Myeng. Visualization of Retrieval Results Using DART. In Proc. of RIAO. Paris, April, 2000. http://133.23.229.11/~ysuzuki/Proceedingsall/RIAO2000/Friday/118BO6.pdf

[2] Oren Zamir and Oren Etzioni Grouper. A Dynamic Cluster Interface to Web Search Results. In Proc. of the Eighth International World Wide Web Conference (WWW8), 1999. http://www.cs.washington.edu/research/projects/WebWare1/etzioni/www/papers/www8.pdf

[3] A. Abdollahzadeh Barfourosh, H.R. Motahary Nezhad, M. L. Anderson and D. Perlis. *Information Retrieval on the World Wide Web and Active Logic: A Survey and Problem Definition*, 2002. http://www.cs.umd.edu/Library/TRs/CS-TR-4291/CS-TR-4291.pdf

[4] Offer Drori. *Improving Display of Search Results in Information Retrieval Systems – Users' Study*. http://shum.huji.ac.il/~offerd/papers/drori072001.pdf

[5] Susan Havre, Elizabeth Hetzler, Ken Perrine, Elizabeth Jurrus, and Nancy Miller. *Interactive Visualization of Multiple Query Result*s. http://www.pnl.gov/infoviz/sparklerinfovis01.pdf

[6] Yahoo http://www.yahoo.com

[7] M. D. Dunlop. Development and Evaluation of Clustering Techniques for Finding People. Proceedings of the Third International Conference on Practical Aspects of Knowledge Management, Basel, Switzerland, 30-31 October 2000. http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-34

[8] M. A. Hearst and J. O. Pedersen. Reexamining the Cluster Hypothesis. In Proceedings of SIGIR '96, pp.76—84, 1996. http://www.scils.rutgers.edu/~muresan/Docs/sigirHearst1996.pdf

[9] Coles, S.L., R.C. DeFelice, and D. Minton. Marine Species Survey of Johnston Atoll, Central Pacific Ocean. Report to U.S. Fish & Wildlife Service, Honolulu. Bishop Museum Technical Report 19, 2001. http://hbs.bishopmuseum.org/pdf/johnstonreport.pdf

[10] Michael Steinbach, George Karypis, and Vipin Kumar. A Comparison of Document Clustering Techniques. Department of Computer Science and Engineering, University of Minnesota, Technical Report #00-034, 2000. http://rakaposhi.eas.asu.edu/cse494/notes/clustering-doccluster.pdf

[11] Douglass R. Cutting, David R. Karger, Jan O. Pedersen, and John W. Tukey. Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections, SIGIR '92, pp.318 – 329, 1992. http://www.scils.rutgers.edu/~muresan/Docs/sigirCutting1992.pdf

[12] Oren Zamir and Oren Etzioni. Web Document Clustering: a Feasibility Demonstration. Proceedings of the 19th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98), pp.46-54, 1998.

[13] Marty Hall and Larry Brown. *Core Web Programming*. New Jersey, Prentice Hall, 2001.

[14] The Java Servlet Technology Pages http://java.sun.com/products/servlet/index.html.

[15] Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman. *Readings in Information Visualization*. Morgan Kaufman, 1999.

[16] Robert Spence. *Information Visualization*. Addison-Wesley, 2001.

[17] David A. Carr. Guidelines for Designing Information Visualization Applications. Proceedings of the 1999 Ericsson Conference on Usability Engineering. http://www.sm.luth.se/~david/papers/VizGuidelines.pdf

[18] Matthew Carey, Frank Kriwaczek, and Stefan M. Ruger. *A Visualization Interface for Document Searching and Browsing*. http://www.doc.ic.ac.uk/~frk/frank/inforet.pdf

PredTOOL – Uma Ferramenta para Apoiar o Teste Baseado em Predicados

Edenilson José da Silva
CEFET/PR Unidade do Sudoeste, GETIC
Pato Branco, Brasil, 85502-390
ede@pb.cefetpr.br

e

Silvia Regina Vergilio
UFPR, Departamento de Informática,
Curitiba, Brasil, 81531-970
silvia@inf.ufpr.br

Abstract

The testing activity is a fundamental phase in the Software Engineering process, especially for improving the quality of the developed programs. To reduce the costs and to increase the number of defects revealed in the test, several testing criteria were proposed. These criteria guide the tester in the selection and evaluation of test case sets. This works focuses on structural testing criteria, more particularly BOR (Boolean OperatoR testing) and BRO (Boolean and Relational Operator testing) criteria, that have the goal of revealing faults in compound predicates of the program under testing. A tool that implements the BOR and BRO criteria is described. This tool, named PredTOOL, supports the test of C programs. PredTOOL made possible the accomplishment of experiments with BOR and BRO criteria and the comparison of those criteria with two other structural criteria: All-edges and All Potential-Uses. The obtained results are used to propose a strategy for application of the studied structural criteria.
Keywords: Software Testing, Predicate Based Testing, Structural Testing Criteria.

Resumo

A atividade de teste é fundamental dentro da Engenharia de Software, especialmente para a melhoria da qualidade dos programas criados. Para reduzir os custos e aumentar o número de defeitos revelados no teste, foram propostos diversos critérios. Esses critérios têm como objetivo guiar o testador na seleção e na avaliação de conjuntos de casos de teste. Este trabalho aborda os critérios estruturais de teste, mais particularmente os critérios BOR (Boolean OperatoR testing) e BRO (Boolean and Relational Operator testing), que têm como objetivo revelar defeitos presentes em predicados compostos do programa em teste. Uma ferramenta que automatiza os critérios BOR e BRO é descrita. Essa ferramenta, chamada PredTOOL permite o teste de programas em linguagem C. A utilização da ferramenta tornou possível a realização de um experimento dos critérios BOR e BRO e a comparação desses critérios com dois outros critérios estruturais, Todos-Arcos e Todos Potenciais-Usos. Da análise dos resultados obtidos, é sugerida uma estratégia para aplicação dos critérios estruturais analisados.
Palavras chaves: Teste de Software, Testes Baseado em Predicados, Critérios de Teste Estrutural**.**

1 - Introdução

A atividade de teste de software é um elemento crítico da garantia de qualidade de software e representa a última revisão das especificações, projeto e codificação [11]. Salienta-se que a atividade de teste tem sido apontada como uma das mais onerosas no desenvolvimento de software.

Nesse contexto, o desenvolvimento de ferramentas para suporte à atividade de teste, é fundamental. Estas ferramentas propiciam maior qualidade e produtividade para a fase de testes, uma vez que essa atividade é muito propensa a erros, além de improdutiva se aplicada manualmente. Estas ferramentas são desenvolvidas baseadas em técnicas e critérios de teste de software. As principais técnicas de teste são: a) Técnica funcional que utiliza aspectos funcionais do software para derivar os dados de teste; b) Técnica estrutural que utiliza o código fonte para derivar os casos de teste e baseia-se no conhecimento da estrutura interna do programa ou da especificação. Geralmente utiliza-se do grafo do programa ou grafo de fluxo de controle (GFC); e c) Técnica baseada em erros, que utiliza defeitos específicos, comuns em programação. A ênfase desta técnica está nos erros que o programador pode cometer durante o desenvolvimento.

As técnicas descritas acima geralmente são associadas a um critério de teste, que são predicados utilizados para se considerar quando a atividade de teste deve ser encerrada, isto é, para considerar que um determinado programa já foi suficientemente testado e ajudar o testador na tarefa de selecionar os casos de teste [7], [14]. Existem vários critérios de teste disponíveis na literatura que consideram diferentes aspectos dos programas ou da especificação para selecionar o conjunto de testes; entre estes critérios são de especial interesse nesse trabalho os critérios estruturais. Geralmente eles requerem a execução de caminhos no programa que devem exercitar elementos do código-fonte ou do grafo de fluxo de controle do programa. Eles podem ser: a) Critérios baseados no fluxo de controle [7], [14]: utilizam características de controle de execução do programa, como nós, arcos e caminhos do grafo de fluxo de controle; dentre estes podemos citar os critérios Todos-nós, Todos-Arcos e Todos-caminhos; b) Critérios baseados no fluxo de dados [7], [14]: exploram a interação que envolve as definições e usos das variáveis no programa. Os principais critérios baseados em fluxo de dados são Todas-Definições, Todos-Usos, Todos-Du-Caminhos e Todos-Potenciais-Usos; e c) Critérios baseados em predicados, que têm o objetivo de descobrir erros nos predicados dos programas ou na especificação. Os critérios BOR (Boolean Operator Testing) e BRO (Boolean and Relational Operator Testing) [15] são exemplos deste tipo de critério. Eles visam a execução de certos tipos de teste para cada predicado (ou condição) do programa, satisfazendo assim um conjunto mínimo de restrições que é criado a partir da análise dos operadores lógico/booleanos do programa em teste.

Os diferentes critérios de teste são considerados complementares pois podem revelar diferentes tipos de teste. Por isso, a realização de estudos empíricos é fundamental para se obter estratégias de aplicação dos critérios de teste existentes.

A aplicação prática dos critérios de teste e a condução de experimentos somente são possíveis se ferramentas de teste estiverem disponíveis. Entre as principais ferramentas destacam-se: a ferramenta Asset [3], para o teste de programas em Pascal e a Atac [5] para programas C. Elas utilizam os critérios de adequação baseados na análise de fluxo de dados definidos por Rapps e Weyuker [13], [14], além de alguns critérios baseados em fluxo de controle. Para os estudos descritos nesse trabalho, utilizou-se a ferramenta Poke-Tool [1], que apóia a utilização da família de critérios Todos Potenciais-Usos, critérios estruturais baseados em fluxo de dados, propostos por Maldonado [7], além dos critérios baseados em fluxo de controle Todos-Arcos e Todos-Nós. Para apoiar a utilização do critério BRO foi desenvolvida na Universidade da Carolina do Norte a ferramenta BGG [15], que apóia o teste de programas escritos em Pascal.

Na literatura encontram-se diferentes trabalhos, e entre esses estudos existem os que comparam os critérios estruturais [2], [7], [16], [18], [19], [20], [21]. Com relação aos critérios BOR e BRO, baseados em predicados, são poucos os relatos desse tipo de estudo. Tai [15] mostra resultados de comparações empíricas desses critérios com o critério Todos-Arcos. Vouk et al [10] comparam o teste baseado em predicados com a estratégia N-versions, mas os critérios baseados em predicados não foram ainda comparados com os critérios baseados em fluxo de dados. Talvez isso se deva ao fato de haver uma única ferramenta para o critérios BOR/BRO para o teste de programas Pascal e a grande maioria de ferramentas para critérios baseados em fluxo de dados apóia o teste de programas C. Apesar disso, experimentos com tais critérios são importantes porque eles podem ressaltar o relacionamento empírico entre os critérios baseados em predicados e outros critérios estruturais.

Dado o contexto descrito acima, esse trabalho tem como objetivo descrever uma ferramenta de apoio ao teste baseado em predicados, ou seja, dos critérios BOR e BRO para programas escritos em linguagem C. Essa ferramenta, chamada PredTOOL, permite a condução de experimentos de comparação e avaliação dos critérios implementados. Assim sendo, um experimento com a PredTOOL e com a ferramenta Poke-Tool, foi realizado. Esse experimento realizou comparações com os critérios BOR e BRO com dois outros critérios estruturais, o critério Todos-Arcos, baseado em fluxo de controle, o critério Todos Potenciais-Usos, baseado

em fluxo de dados. Os resultados mostram uma relação empírica que pode ser utilizada para se propor uma estratégia de utilização dos critérios estudados.

O trabalho está organizado da seguinte maneira. Na Seção 2 é apresentada uma revisão do teste estrutural de software e dos critérios utilizados. Na Seção 3 é descrita a ferramenta PredTOOL. Na Seção 4 são discutidos os resultados do experimento realizado. Na Seção 5 estão as conclusões e desdobramentos desse trabalho.

2 –Teste Estrutural de Software

Glen Myers [9] estabelece três regras que podem ilustrar os objetivos de teste: 1) A atividade de teste é o processo de executar um programa com a intenção de descobrir um erro; 2) Um bom caso de teste é aquele que tem uma elevada probabilidade de revelar um erro ainda não descoberto; 3) Um caso de teste é composto de uma entrada para o programa e da saída esperada; e 4) Um teste bem sucedido é aquele que revela um erro ainda não descoberto.

Duas questões são importantes na atividade de teste e devem ser levadas em consideração quando os testes são realizados: "Como os dados de teste devem ser selecionados?" e "Como decidir se um programa P foi suficientemente testado?". Levando-se em consideração estas questões, têm-se que os critérios para selecionar e avaliar conjuntos de casos de teste são fundamentais para o sucesso da atividade de teste [7], [14]. Os principais critérios para geração de dados de teste visam gerar dados que possam detectar a maioria dos defeitos com o mínimo de esforço e tempo. Nessa seção os critérios de teste estruturais utilizados nesse trabalho são descritos. Esses critérios consideram aspectos da estrutura interna do programa ou especificação para derivar os dados de teste. Os critérios estruturais mais conhecidos são:

1) Critérios baseados em fluxo de controle: Utilizam apenas características de controle da execução do programa, e os critérios mais importantes são: Todos-Nós [12]; Todos-Arcos ou Todos-Ramos [12], e Todos-Caminhos [4], [12] que exigem respectivamente que todos os nós, arcos e caminhos do GFC (grafo de fluxo de controle) sejam exercitados pelos casos de teste.
2) Critérios baseados em fluxo de dados: Utilizam informações do fluxo de dados do programa para determinar os requisitos de teste, selecionando caminhos de teste de acordo com as localizações das definições e usos de variáveis no programa. Os principais representantes deste tipo de critérios são: Todas-Definições; Todos-Usos; Todos-Potenciais-Usos [7], [8], [14]. O critério Todos-Potenciais-Usos requer a execução de caminhos para cobrir associações entre um nó do grafo i que define uma variável x, com outros nós do grafo que potencialmente usam x.
3) O teste baseado em predicados consiste na execução de determinados tipos de testes para cada predicado (ou condição) do programa. Os critérios BOR (Boolean OperatoR testing) e BRO (Boolean and Relational Operator testing) [15], são os mais conhecidos critérios baseados em predicados e são o foco desse trabalho. Por isso, eles serão descritos em detalhe na Subseção 2.1.

O teste estrutural apresenta um problema, que refere-se a impossibilidade de se determinar automaticamente se um dado caminho é ou não executável. Assim, não existe um algoritmo que, dado um caminho completo qualquer, decida se o caminho é executável e forneça o conjunto de valores que causam a execução desse caminho [17]. É preciso a intervenção do testador para determinar quais são os caminhos não executáveis para o programa sendo testado. Um elemento requerido por um dado critério estrutural será não executável se não existir um caminho executável que o cubra. Com esse problema, uma grande quantidade de tempo/esforço é gasta determinando-se elementos não executáveis na aplicação dos critérios estruturais.

2.1 – Critérios BOR/BRO

Predicados (ou condições) em um programa dividem o domínio de entrada deste programa em partições e definem os caminhos deste programa. Seja C um predicado de uma sentença *if* ou *while* em um programa P. Assume-se que a execução de P com a entrada X atinge C. Se o resultado de C é incorreto devido a erros em C ou na(s) sentença(s) executada(s) antes de alcançar C, então um caminho incorreto de P será executado e provavelmente um resultado incorreto será produzido. Através do teste de C é possível detectar não somente erros em C ou nas instruções executadas antes de atingi-lo, mas erros nas instruções executadas após C.

Um predicado em um programa ou é um predicado simples ou então é um predicado composto. Um predicado simples é uma variável booleana ou uma expressão relacional, possivelmente com um ou mais operadores de negação. Uma expressão relacional tem a forma:

E1 <rop> E2

onde E1 e E2 são expressões aritméticas e <rop> é um dos 6 operadores relacionais: "<", "≤", "=", "≥", ">", e "≠". Um predicado composto consiste de pelo menos um operador binário booleano, dois ou mais operandos, possivelmente operadores de negação e parênteses.

Os operadores booleanos que podem estar presentes em um predicado são OU ("||") e ("&&"), cada um com dois operandos. Um operando simples em um predicado composto refere-se a um operando sem operadores binários booleanos. Um operando composto em um predicado refere-se a um operando com pelo menos um operador binário boleano. Uma expressão booleanas é um predicado sem expressões relacionais.

Para formalizar a ocorrência de variáveis e operadores nas expressões apresentadas neste artigo, será utilizada a representação abaixo descrita:

- Bi com i > 0 denota uma variável booleanas,
- Ei uma expressão aritmética;
- <rop> ou <ropi> um operador relacional e
- <bop> ou <bopi> um operador binário boleano.

Para um predicado com n, n > 0 operandos simples,

- (<opd1> <bop1> <opd2> ... ... <bopn-1> <opdn),

onde <opdi>, i > 0, denota o i-ésimo operando simples, uma restrição-BR (ou somente restrição) é definida como (D1, D2,.... Dn), onde Di, $0 < i \leq n$, é um símbolo especificando uma restrição na variável booleanas ou expressão relacional em <opdi>. Observa-se que um operando simples é uma variável booleana ou expressão relacional, possivelmente com um ou mais operadores de negação.

Para uma variável booleana B, os seguintes símbolos são usados para denotar diferentes tipos de restrições no valor de B:

- t: o valor de B é verdadeiro;
- f: o valor de B é falso;
- *: Não há restrição sobre o valor de B;

Para uma expressão relacional E1 <rop> E2, os seguintes símbolos são usados para denotar diferentes tipos de restrições sobre os resultados da expressão relacional:

- t: o valor da expressão relacional é verdadeiro;
- f: o valor da expressão relacional é falso;
- >: o valor de (E1 - E2) é maior do que zero;
- =: o valor de (E1 - E2) é igual a zero;
- <: o valor de (E1 - E2) é menor do que zero;
- $+\varepsilon$: o valor de (E1 - E2) é maior do que zero e menor ou igual a ε;
- $-\varepsilon$: o valor de (E1 - E2) é menor do que zero e maior ou igual a ε;
- *: não há restrição no valor da expressão.

Uma restrição D para um predicado C é coberta (ou satisfeita) por um teste se durante a execução de C com este teste, o valor de cada variável booleana ou expressão relacional em C satisfaz a restrição correspondente em D. Considere a restrição (=,<) para o predicado ((E1 > E2) || ~ (E3 > E4)), onde ~ é o operador de negação. A cobertura de (=,<) para este predicado requer um teste fazendo E1 = E2 e E3 < E4. A cobertura de (t, $+\varepsilon$) para este predicado requer um teste fazendo E1 > E2 e $0 < E3 - E4 \leq \varepsilon$.

Um conjunto S de restrições para um predicado C é dito ser coberto (ou satisfeito) por um conjunto de teste T se cada restrição em S é satisfeita em C por pelo menos um teste em T. Um teste em T pode cobrir duas ou mais restrições em S.

Os critérios BOR e BRO requerem basicamente um conjunto de restrições para os predicados do programa em teste e que um conjunto de casos de teste T cubra as restrições requeridas.

O algoritmo, extraído de [15], para gerar restrições requeridas para os critérios BOR e BRO é apresentado abaixo.

Sejam u = (u1,..., um) e v = (v1,..., vn), onde m, n > 0, são duas listas de elementos. A concatenação de u e v, denotada (u, v), é (u1,..., um, v1,..., vn). Sejam A e B dois conjuntos de listas. $A \cup B$ denota a união de A e B, A*B o produto de A por B, e |A| o tamanho de A. Já A % B é chamada de "onto" de A para B, sendo o conjunto mínimo de (u, v) tal que u|v está em A|B e cada elemento em A|B aparece com, o u|v pelo menos uma vez. Em outras palavras, A % B é um conjunto mínimo de (u,v) tal que u e v estão em A e B respectivamente, cada elemento de A aparece como u pelo menos uma vez, e cada elemento de B aparece como v pelo menos uma vez. |A % B| é o máximo entre |A| e |B|. Se ambos A e B têm mais do que dois elementos, A % B pode formar diversos conjuntos e retorna qualquer um deles. Por exemplo, considere C = {(a), (b)} e D = {(c), (d)}. C % D tem dois valores possíveis: {(a,c),(b,d)} e {(a,d),(b,c)}. Seja E = {(a), (b)} e F = {(c),(d),(e)}, E % F tem 6 valores possíveis: {(a,c),(b,d),(a,e)}, {(a,c),(b,d),(b,e)}, {(a,c),(a,d),(b,e)}, {(b,c),(a,d),(b,e)}, {(b,c),(a,d),(a,e)}, {(b,c),(b,d),(a,e)}.

Seja X uma restrição, que é formada por valores "t", "f", ">", "=" e "<" para um predicado C, o valor produzido por C em qualquer entrada que satisfaça X é o mesmo, e é chamado de C(X) - então uma restrição para C pode ser vista como uma entrada de C. X cobre ou está associado com o ramo verdadeiro ou falso de C se C(X) = verdadeiro ou C(X) = falso. Seja S um conjunto de restrições para C. S pode ser dividido em dois conjuntos: S_t e S_f, onde S_t (C) = {X está em S | C(X) = t} e S_f (C) = {X está em S | C(X) = f}.

Seja w uma entrada que satisfaz X para C. O valor produzido por C com w é C(X). Seja C" um predicado que tem o mesmo conjunto de variáveis de entrada que C. O valor produzido por C" sobre a entrada w pode ser chamado de C"(X) sob uma das duas condições:

1 - C" difere de C somente nos operadores booleanos e cada restrição em X é ou "t" ou "f", e

2 - C" difere de C somente nos operadores booleanos e relacionais e cada restrição em X é ou "t" ou "f", para uma variável booleana em C, ou ">", "=", ou "<" para cada expressão relacional em C.

Para uma variável booleana, seu conjunto de restrições BOR ou BRO é definido como {(t),(f)}. Para uma expressão relacional (E1 <rop> E2), seu conjunto de restrições BOR é definido como {(t),(f)} e seu conjunto de restrições BRO como {(<), (=), (>)}. Sejam C1 e C2 predicados. S1 e S2 são os conjuntos de restrições BOR (BRO) para C1 e C2 respectivamente. A seguir é mostrado como construir os conjuntos de restrições BOR (BRO) para (C1 || C2) e (C1 && C2) usando S1 e S2. Na Tabela 2.1 são mostradas as restrições possíveis para uma expressão relacional R. Sejam S1 e S2 conjuntos de restrições BOR (BRO) para C1 e C2, respectivamente, para gerar os conjuntos BOR e BRO aplicam-se as seguintes regras.

Regra 1: Para C = (C1 || C2)

F(C) = S1_f % S2_f e T(C) = {S1_t * {f2}} $ {{f1} * S2_t}
Onde: f1 ∈ S1_f e f2 ∈ S2_f e (f1, f2) ∈ F(C). Assim, T(C) ∪ F(C) é um conjunto de restrições BOR(BRO) para C.

Regra 2: Para C = (C1 && C2)

T(C) = S1_t % S2_t e F(C) = {S1_f * {t2}} $ {{t1} * S2_f }
Onde: t1 ∈ S1_t e t2 ∈ S2_t e (t1, t2) ∈ T(C). Assim, T(C) ∪ F(C) é um conjunto de restrições BOR(BRO) para C.

Tabela 2.1- Restrições Para a Expressão Relacional R

Operador Relacional	Restrições para os predicados	
	S_t(R)	S_f(R)
=	{ = }	{<,>}
≠	{<,>}	{ = }
>	{ > }	{<, =}
<	{ < }	{=, >}
≥	{>, =}	{ < }
≤	{=, <}	{ > }

Exibe-se, na seqüência, a construção do conjunto de restrições BOR para o predicado composto C# definido como (E1 = E2) && (E3 ≠ E4).

Sejam C1 e C2 as expressões (E1 = E2) e (E3 ≠ E4), nesta ordem. Para a construção do teste BOR para C1 e C2, S1_t = S2_t = {(t)} e S1_f = S2_f = {(f)}. De acordo com a Regra 2, temos que T(C#) = S1_t % S2_t = {(t,t)}. Uma vez que t1 = t2 = t, F(C#) = {S1_f * {t2}} ∪ {{t1} * S2_f} = {(f,t),(t,f)}. Assim, {(t,t), (f,t),(t,f)} é um conjunto de restrições BOR para C#. Observa-se que para a expressão (J1 && J2), onde J1 e J2 representam variáveis booleanass distintas, tem o mesmo conjunto de restrições BOR que C#.

Exibe-se também, a construção do conjunto de restrições BRO para C#. Para o teste BRO de C1, S1_t = {(=)} e S1_f = {(>),(<)}. No caso de C2, S2_t = {(>),(<)} e S2_f = {(=)}. Seguindo a Regra 2, temos que T(C#) = S1_t % S2_t = {(=)} % {(>),(<)} = {(=,>), (=,<)}. Como T(C#) tem dois elementos, escolhe-se (=, >) como (t1, t2). Deste modo, F(C#) = {S1_f * {>}} ∪ {{=} * S2_f} = {(>,>),(<,>),(=,=)}. Desta forma, {(=,>),(=.<),(>,>).(<,>),(=,=)} é um conjunto de restrições BOR para C#. Para o predicado C@, denotado por ((E1 < E2) && ((E3 > E4) || (E5 = E6))), o conjunto de restrições BRO é construído considerando-se a árvore sintática da Figura 2.1.

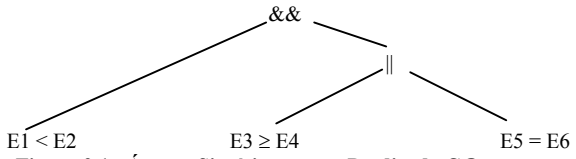


Figura 2.1 - Árvore Sintática para o Predicado C@.

Os seguintes passos são seguidos:

(1) Gerar o conjunto de restrições BRO S1, S2 e S3 para (E1 < E2), (E3 ≥ E4) e (E5 = E6).

(2) Aplicar a regra 1 em S2 e S3 e construir um conjunto de restrições S4 para ((E3 ≥E4) || (E5 = E6)).

(3) Aplicar a regra 2 sobre S1 e S4 e construir o conjunto de restrições S5 para C@.

Na Figura 2.2, o processo de aplicação dos passos descritos acima é exibido com os conjuntos de restrições ao predicado C@.

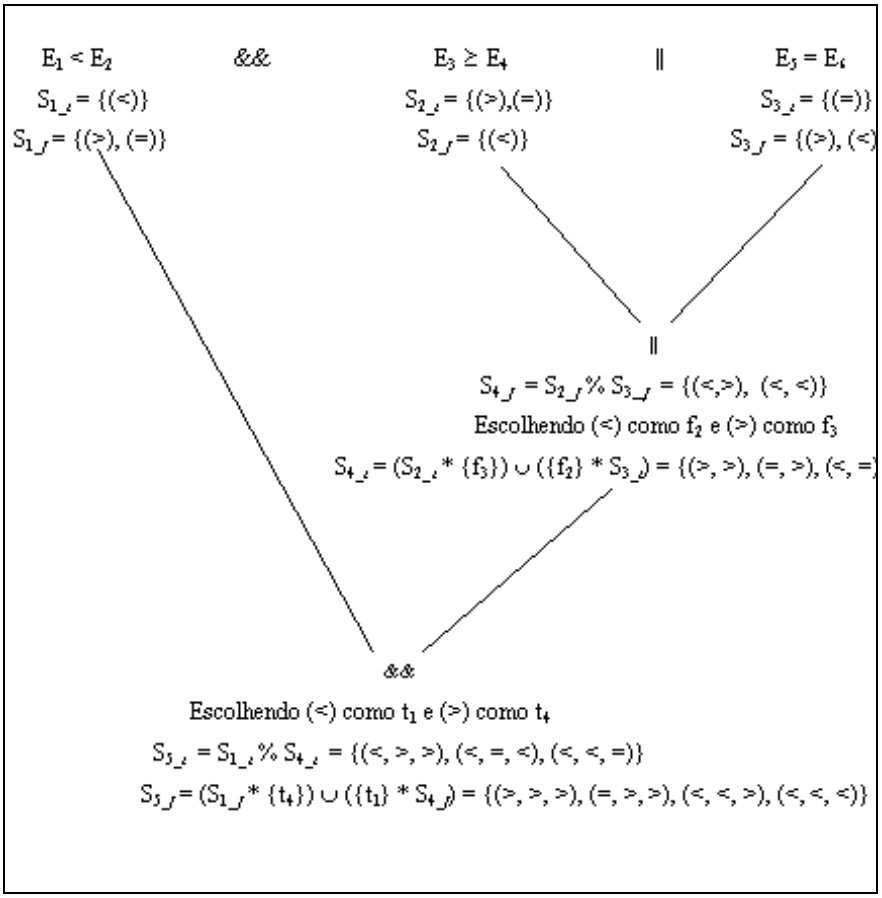


Figura 2.2 - Restrições BRO para o predicado ((E1 < E2) && ((E3 > E4) || (E5 = E6))),

Como os predicados aparecem na especificação e na implementação de um programa, os critérios BOR e BRO podem ser usados nos testes baseados em programas e nos testes baseados em especificações. Aplicar um dos critérios a um programa P envolve: 1) A geração do conjunto de restrições BOR (BRO) para cada predicado em P e; 2) A geração de testes de P para cobrir cada uma destas restrições pelo menos uma vez.

Um caminho de P pode ser definido como uma seqüência de ramos em P. Seja um caminho-restrição de P um caminho de P onde cada ramo é substituído por uma restrição. Uma maneira de executar o passo 2 é: 2.1) - Selecionar um conjunto de caminhos - restrições de P para cobrir cada uma das restrições geradas em (1) pelo menos uma vez e; 2.2) - Gerar um teste para cada caminho-restrição selecionada.

Este método consome tempo, pois existem poucas ferramentas para seleção automática de caminhos e testes. Uma alternativa é testar P com um conjunto de testes existente T, que pode conter testes gerados aleatoriamente a partir do domínio de entrada de P. Durante ou após o teste de P com T, as restrições satisfeitas por T são determinadas. Portanto, as restrições não satisfeitas em P são usadas para guiar a seleção de caminhos - restrições e a geração de testes adicionais para P.

Uma dificuldade maior no teste de software é que um caminho selecionado de um programa pode ser não executável; ou seja: não existem entradas para executar o caminho. Além disso, o problema de determinar se um caminho é executável é geralmente indecidível. Tais dificuldades também existem para caminhos - restrições. Um caminho-restrição é não executável devido a uma ou mais restrições no caminho.

Uma restrição para o predicado C é não executável para C se ela nunca pode ser satisfeita por um teste para C. Por exemplo, a restrição (t, t) é não executável para o predicado ((E1 > E2) || (E1 = E2)), já que o valor de E1 nunca pode ser ao mesmo tempo maior do que E2 e igual a E2. Se todos os operandos simples em um predicado C são independentes uns dos outros (ou seja: sem variáveis comuns), então cada restrição para C poderá ser executável. Assuma que P contém a seguinte sentença:

if (X > Y) then if ((X <= Z) || (Z > Y)) then ...;

A restrição (<, <) é executável para ((X <= Z) || (Z > Y)) e sua cobertura requer um teste fazendo X < Z < Y. Contudo, tal teste nunca irá alcançar ((X <= Z) || (Z > Y)), pois ele não pode satisfazer o predicado X > Y. Logo, a restrição (<, <) para ((X <= Z) || (Z > Y)) é não executável para P.

Se alguma restrição gerada para atender aos critérios BOR e BRO for não executável, então é impossível cobrir todas as restrições geradas e não há garantia para a detecção de erros nos operadores booleanos e relacionais em um programa. Atenta-se para o fato de que o problema de determinar se uma restrição para um predicado é não executável é, em geral, indecidível [15]; problema análogo aos elementos não executáveis requeridos pelos outros critérios estruturais

3 – A Ferramenta PredTOOL

A Figura 3.1 mostra a arquitetura da ferramenta, composta por módulos que se comunicam através de arquivos. Nela, os retângulos representam os módulos, os losangos representam as entradas fornecidas pelos usuários e os círculos os produtos gerados. A seguir, tem-se uma breve descrição de cada módulo:

1) MÓDULO GERADOR – Identificação dos predicados: Esta função deve identificar os predicados da unidade a ser testada. A PredTOOL utiliza uma linguagem intermediária para localizar no código-fonte a exata posição onde encontram-se os predicados (condições) do programa, salvando os mesmos em arquivo. Com estas informações pode-se então passar para o próximo passo, a instrumentação.

2) INSTRUMENTAÇÃO - A instrumentação insere comando de escrita, chamados de pontas de prova, no programa fonte. Tais comandos basicamente inserem no programa todas as combinações possíveis de restrições aplicadas a cada predicado. O objetivo é gerar uma nova versão do programa - a versão instrumentada - para produzir um "trace" da execução dos casos de teste e obter a informação de quais restrições (do conjunto total de restrições possíveis) foram executadas com os casos de teste. É importante garantir que o programa instrumentado tenha o mesmo comportamento do programa em teste, ou seja, a semântica do programa em teste não é afetada pela instrumentação.

3) GERAÇÃO DO EXECUTÁVEL - O programa instrumentado deve então ser compilado pelo testador para gerar o executável da versão instrumentada. Este arquivo será então utilizado para receber como entrada os casos de teste desejados para verificar a cobertura que será atingida.

4) EXECUÇÃO DOS CASOS DE TESTE - Com o programa executável obtido na etapa anterior, deve-se proceder a execução da unidade em teste, produzindo o conjunto de restrições executadas pelos casos de teste fornecidos pelo usuário. Os casos de teste fornecidos são também armazenados em arquivos.

5) AVALIADOR – Este módulo verifica se o conjunto de restrições executadas pelos casos de teste satisfaz os critérios BOR/BRO. Em caso negativo, produz uma relação das restrições requeridas pelo critério e não executadas. Uma medida percentual da cobertura provida pelo conjunto de casos de teste também é gerada, isto é, uma relação entre as restrições executadas e as restrições requeridas é produzida. Entre as restrições requeridas, podem existir restrições não executáveis. A PredTOOL, atualmente, não fornece nenhum suporte para determinação da executabilidade de uma determinada restrição, por ser este um problema indecidível e de difícil solução. O usuário deverá determiná-la manualmente.

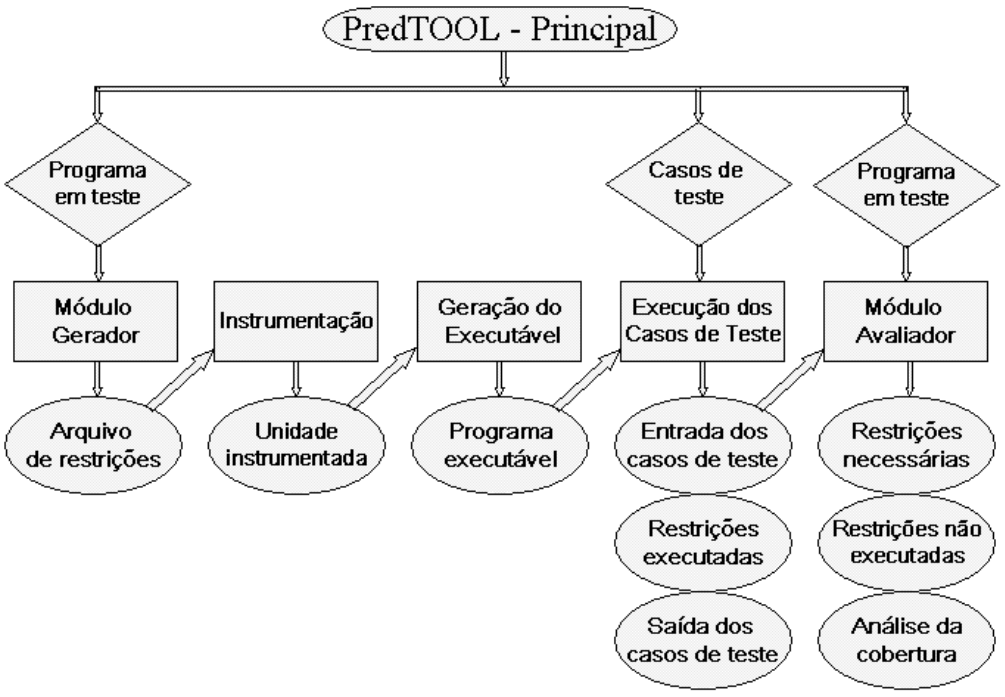


Figura 3.1 Principais Módulos da Ferramenta PredTOOL

Para iniciar o uso da PredTOOL, deve-se especificar na linha de comando de execução da ferramenta, o arquivo para ser realizada a geração das restrições como mostrado na Figura 3.2.

Deve-se informar o nome do arquivo sem extensão, pois a PredTOOL automaticamente reconhecerá o arquivo fonte para iniciar o processo de criação das restrições requeridas. Na figura também pode-se visualizar as restrições requeridas pelo predicado do programa utilizado como exemplo. A execução inicia com a leitura dos parâmetros do arquivo com o programa em linguagem intermediária e baseando-se nos parâmetros contido neste arquivo, o programa lê a expressão no arquivo fonte e repete esta operação até que todas as expressões tenham sido lidas e armazenadas.

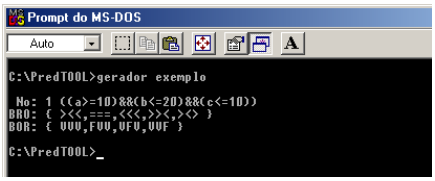


Figura 3.2 – Geração das Restrições Requeridas Pela PredTOOL.

Após a leitura das expressões, inicia-se a geração das restrições BOR/BRO para cada uma das expressões. A ferramenta faz a leitura da primeira expressão, traduzindo-a numa árvore binária gramatical e percorre a árvore fazendo a geração das restrições nó a nó, em pós-ordem. No final da geração, o nó raiz conterá as restrições para toda a expressão. Então, estas restrições são gravadas em arquivo e uma nova expressão é então lida. Esse ciclo termina quando forem geradas restrições para todas as expressões dos nós do programa.

Ao final desta etapa, o arquivo com as restrições requeridas é criado e pode ser visto na Figura 3.3. Os valores dos campos de cada linha deste arquivo são separados pelo símbolo ';'.

```
1;((a>=10)&&(b<=20)&&(c<=10));1;1;,1 1 1 ,;1;3;,0 1 1 ,1 0 1 ,1 1 0 ,;1;2;6 7 7 ,10 10 10 ,;1;3;7 7 7 ,6 6 7 ,6 7 6 ,;
```

Figura 3.3 –Conteúdo do arquivo de restrições do exemplo utilizado.

O valor do primeiro campo representa o nó da expressão; o segundo campo mostra a expressão de onde será gerada as restrições requeridas. O terceiro campo tem a função de indicar a existência ou não das restrições para o critério BOR ou BRO, caso o testador tenha optado por qualquer um dos critérios ou ambos. Na versão apresentada, a PredTOOL sempre gera as restrições para ambos os critérios, desta forma, este campo terá sempre o valor igual a 1. O próximo campo indica a quantidade de restrições existentes para o critério e o último campo mostra as restrições requeridas.

Nesta etapa também é gerado o arquivo que contém as pontas de prova, chamado de arquivo instrumentado. Esse arquivo gerado deve então ser compilado para obter o executável que receberá os casos de teste.

Com o programa executável, deve-se então proceder a entrada os casos de teste para tentar cobrir todas as restrições requeridas. Note-se que neste processo podem existir restrições não executáveis, e cabe ao testador conferir esta situação. Por este motivo, a cobertura de 100% das restrições pode não ser alcançada. Importante ressaltar que a inclusão de novos casos de teste não sobrescrevem casos de teste anteriores, os novos casos são adicionados no conjunto para análise da cobertura.

A Tabela 3.1 mostra os dois casos de teste inseridos no exemplo descrito.

Tabela 3.1 – Detalhamento dos Arquivos de Testes Submetidos no Exemplo.

Número do caso de teste	Nome do caso de teste	Valores digitados para as variáveis		Arquivo Gerado	Conversão do arquivo gerado nas restrições	Significado do arquivo gerado
		C	X			
01	exemplo_pp1.pp	5	10	1; 7,1 6,0 6,1	1; <,T >,F >,T	Nó 1; Restrições BRO: <>>; Restrições Bor = TFT
02	exemplo_pp2.pp	1	1	1; 7,1 7,1 7,0	1; <,T <,T <,F	Nó 1; Restrições BRO: <<<; Restrições Bor = FTT

Os valores do arquivo gerado seguem a seguinte legenda:

0 = F (false);
1 = T (true);
6 = > (maior);
7 = < (menor);
10 = = (igual).

A ordem em que o arquivo é gerado deve ser invertida para que as restrições encontrem-se na ordem correta, isto devido a execução da função pp (ponta de prova) do programa instrumentado, que gera os dados na ordem de precedência das operações. Assim, para a correta geração das restrições elas devem ser invertidas na ordem em que o arquivo é gerado.

Ao se executar o programa instrumentado, os traces correspondentes a cada dado de teste serão gerados e posteriormente utilizados na avaliação.

O módulo AVALIADOR utiliza-se dos arquivos gerados nas etapas anteriores para realizar a avaliação das restrições e apresentar a cobertura atingida com os casos de teste inseridos. Deve-se informar o nome do

arquivo sem extensão; assim a PredTOOL irá automaticamente identificar os arquivos criados nas etapas anteriores e realizar a avaliação dos arquivos gerados. Caso algum arquivo esteja faltando, uma mensagem será disparada avisando o testador. Caso tudo ocorra normalmente a Figura 3.4 mostra o arquivo final onde estão os casos de (Tabela 3.1), a cobertura de cada um, as restrições requeridas e executadas, as restrições requeridas e não executadas e a cobertura obtida:

```
Caso de Teste:  exemplo_pp1.pp
No: 1  ((a>=10)&&(b<=20)&&(c<=10))
Total de Cobertura das Restrições BRO: 20.00%
Total de Cobertura das Restrições BOR: 25.00%

Caso de Teste:  exemplo_pp2.pp
No: 1  ((a>=10)&&(b<=20)&&(c<=10))
Total de Cobertura das Restrições BRO: 20.00%
Total de Cobertura das Restrições BOR: 25.00%
----------------------------------------
Total de Cobertura das Restrições:
----------------------------------------
No: 1  ((a>=10)&&(b<=20)&&(c<=10))

Total de Cobertura das Restrições BRO: 40.00%
{ (>><)  (<<<) }
Restrições BRO Não Cobertas: 60.00%
{ (><<)  (===)  (><>) }
Total de Cobertura das Restrições BOR: 50.00%
{ (TFT)  (FTT) }
Restrições BOR Não Cobertas: 50.00%
{ (TTT)  (TTF) }
```

Figura 3.4 – Arquivo de exemplo com avaliação final da PredTOOL do programa testado.

4 – Experimento

Apresenta-se um experimento que compara os critérios baseados em predicado, implementados pela PredTOOL - BOR/BRO - com dois outros critérios implementados pela POKE-TOOL - Todos Potenciais-Usos (PU - critério baseado em fluxo de dados) e Todos-Arcos (ARCS - critério baseado em fluxo de controle). O experimento utilizou 6 programas escritos na linguagem C, alguns deles extraídos de [6] e primeiramente utilizados por Weyuker [19].

4.1 – Descrição do Experimento

A descrição funcional dos programas escolhidos encontra-se na Tabela 4.1.

Tabela 4.1 - Descrição Funcional dos Programas Utilizados no Experimento.

Programa	Descrição do Programa
bbsort.c	Programa que utiliza o método de bolha de ordenação. É solicitado a quantidade de números a serem ordenados, apresentando ao final os números ordenados.
compress.c	Encurta uma string padronizando a repetição de caracteres, substituindo a seqüência de quatro ou mais caracteres iguais por ~Nx. N corresponde à posição da letra no alfabeto para uma repetição de x. Grupos maiores que 26 são quebrados em 2.
entab.c	Substitui strings de brancos por tabs, produzindo a mesma saída visual, mas com menos caracteres.
expand.c	Com a entrada padronizada pelo programa COMPRESS.C na forma ~Nx descrita acima, este programa retorna a entrada na sua forma normal,
find.c	Permuta os elementos de um array de forma que todos os elementos à esquerda do índice serão menores ou iguais a este e os elementos à direita serão maiores ou iguais ao índice.
getcmd.c	Decodifica o tipo de comando digitado.

O experimento consistiu dos seguintes passos:

1º – Gerar um conjunto inicial ad-hoc de testes T baseado na funcionalidade de cada programa;

2º – Submeter T na ferramenta de teste Poke-TOOL;

3º – Verificar a cobertura obtida com T para o critério mais fraco, ARCS, e após, para o critério PU. Para ambos os critérios, identificar os elementos não executáveis e gerar os casos de teste adicionais para cobrir todos os elementos executáveis. As Tabelas 4.2 e 4.3 mostram para cada programa o número de elementos requeridos, de elementos não executáveis encontrados, a cobertura obtida e o número de casos de teste efetivos, ou seja, o número de casos de testes que realmente contribuíram para o aumento da cobertura. Os conjuntos TARCS e TPU, que correspondem respectivamente aos conjuntos ARCS adequado e PU adequado são compostos somente de casos de teste efetivos.

4º – Submeter o conjunto de testes T na PredTOOL e gerar os casos de testes adicionais para obter os conjuntos TBOR e TBRO, respectivamente, BOR e BRO adequados. Os resultados encontram-se nas Tabelas 4.4 e 4.5.

5º – Submeter os conjuntos TARCS e TPU, na PredTOOL, para obter a cobertura para os critérios BOR e BRO. As Tabelas de 4.6, 4.7, 4.8 e 4.9 mostram os resultados obtidos.

6º – Submeter os conjuntos TBOR e TBRO na POKE-TOOL para obter a cobertura dos critérios ARCS e PU. As Tabelas 4.10, 4.11, 4.12 e 4.13 mostram os resultados deste passo.

Tabela 4.2 - Resultados da Aplicação de Todos-Arcos.

CRITÉRIO ARCS - RESULTADOS				
Programa	Requeridas	Não executáveis	(TPU)	Cobertura PU (em %)
bbsort.c	7	0	3	100,0
compress.c	10	0	3	100,0
entab.c	11	0	4	100,0
expand.c	7	0	3	100,0
find.c	13	0	4	100,0
getcmd.c	15	0	15	100,0
Total	63	0	32	100,0

Tabela 4.3 - Resultados Todos Potenciais-Usos.

CRITÉRIO PU - RESULTADOS				
Programa	Requeridas	Não executáveis	Efetivos (TPU)	Cobertura PU (em %)
bbsort.c	53	1	6	98,2
compress.c	51	3	10	98,1
entab.c	87	13	17	78,5
expand.c	44	14	9	68,1
find.c	175	51	14	83,7
getcmd.c	15	0	15	100,0
Total	425	82	71	80,7

Tabela 4.4 - Resultados da Aplicação do Critério BOR.

CRITÉRIO BOR – RESULTADOS				
Programa	Requeridas	Não executáveis	Efetivos	Cobertura (em %)
bbsort.c	10	0	2	100,0
compress.c	20	0	3	100,0
entab.c	18	0	4	100,0
expand.c	12	0	4	100,0
find.c	21	0	3	100,0
getcmd.c	30	0	16	100,0
Total	111	0	32	100,0

Tabela 4.5 - Resultados da Aplicação do Critério BRO.

CRITÉRIO BRO – RESULTADOS				
Programa	Requeridas	Não executáveis	Casos de teste	Cobertura (em %)
bbsort.c	15	01	04	93,4
compress.c	30	07	07	72,7
entab.c	27	06	05	77,8
expand.c	18	04	04	77,8
find.c	32	02	05	93,7
getcmd.c	45	01	16	97,8
Total	167	21	41	87,4

Tabela 4.6 - TARCS na PredTOOL. – Critério BOR

CRITÉRIO BOR – RESULTADOS COM TARCS		
Programa	Restrições Cobertas	Cobertura a (em %)
bbsort.c	10	100,0
compress.c	19	95,0
entab.c	17	94,5
expand.c	12	100,0
find.c	21	100,0
getcmd.c	29	96,7
Total	108	97.7

Tabela 4.7 - TARCS na PredTOOL - Critério BRO.

CRITÉRIO BRO – RESULTADOS COM TARCS		
Programa	Restrições Cobertas	Cobertura a (em %)
bbsort.c	10	60,0
compress.c	17	56,7
entab.c	18	66,7
expand.c	13	72,3
find.c	27	84,4
getcmd.c	41	66,7
Total	126	75,4

Tabela 4.8 - TPU na PredTOOL - Critério BOR.

CRITÉRIO BOR – RESULTADOS COM TPU		
Programa	Restrições Cobertas	Cobertura (em %)
bbsort.c	10	100,0
compress.c	20	100,0
entab.c	17	94,5
expand.c	13	100,0
find.c	21	100,0
getcmd.c	29	96,7
Total	109	98,1

Tabela 4.9 - TPU na PredTOOL - Critério BRO.

CRITÉRIO BRO – RESULTADOS COM TPU		
Programa	Restrições Cobertas	Cobertura (em %)
bbsort.c	13	100,0
compress.c	17	100,0
entab.c	19	94,5
expand.c	13	100,0
find.c	30	100,0
getcmd.c	41	96,7
Total	133	79,6

Tabela 4.10 - TBOR na POKE-TOOL - Critério ARCS.

CRITÉRIO ARCS– RESULTADOS COM TBOR		
Programa	Restrições Cobertas	Cobertura (em %)
bbsort.c	7	100,0
compress.c	10	100,0
entab.c	11	100,0
expand.c	7	100,0
find.c	13	100,0
getcmd.c	15	100,0
Total	63	100,0

Tabela 4.11 - TBOR na POKE-TOOL - critério PU.

CRITÉRIO PU – RESULTADOS COM TBOR		
Programa	Restrições Cobertas	Cobertura (em %)
bbsort.c	28	65,1
compress.c	39	76,4
entab.c	40	45,9
expand.c	14	31,8
find.c	84	48,0
getcmd.c	15	100,0
Total	220	51,7

Tabela 4.12 - TBRO na POKE-TOOL - Critério ARCS.

CRITÉRIO ARCS – RESULTADOS COM TBRO		
Programa	Restrições Cobertas	Cobertura (em %)
bbsort.c	7	100,0
compress.c	10	100,0
entab.c	11	100,0
expand.c	7	100,0
find.c	13	100,0
getcmd.c	15	100,0
Total	63	100,0

Tabela 4.13 - TBRO na POKE-TOOL - Critério PU.

CRITÉRIO PU – RESULTADOS COM TBRO		
Programa	Restrições Cobertas	Cobertura (em %)
bbsort.c	31	72,9
compress.c	41	80,4
entab.c	47	54,0
expand.c	26	65,0
find.c	105	60,0
getcmd.c	15	100,0
Total	265	62,3

4.2 – Análise dos Resultados

Os resultados são analisados de acordo com os fatores custo (obtido pelo número necessário de casos de teste para cobrir as restrições) e strength (relacionado com a dificuldade de aplicação) [21].

4.2.1– Custo

A POKE-TOOL não separa as condições nas declarações de fluxo de controle para gerar os arcos requeridos, é suficiente executar as condições *else* e *true* de cada condição. Em alguns casos, quando as expressões não contém predicados compostos, os critérios ARCS e BOR são muito similares. Talvez por causa disto, os critérios requerem exatamente o mesmo número total de casos de teste, 32, muito embora, BOR exija a execução de quase duas vezes mais elementos (restrições). BRO requer 41; estes números de elementos requeridos e casos de teste necessários, não são significativamente maiores do que BOR.

Observou-se que o critério mais exigente é PU. Ele requer 71 casos de teste, mais de duas vezes o número de casos de teste requeridos por ARCS e BOR. Observou-se também que o programa getcmd requer um grande número de casos de teste para os critérios ARCS, BOR e BRO, mas não para PU. Este fato é explicado pela análise dos programas e da complexidade de McCabe [11]. O programa *getcmd.c* tem a maior complexidade. ARCS, BOR e BRO são critérios associados com as condições nos comandos de desvio de fluxo de controle, tais como if, while, case, etc. Assim, maior o número de comandos de desvio de fluxo de controle, isto é, a complexidade de McCabe, maior o número de elementos requeridos por estes critérios e conseqüentemente maior o número de casos de teste necessários. Entretanto, PU é um critério baseado em fluxo de dados, então, existem outras características que influenciam o número de elementos requeridos. Talvez, por causa disto, o programa que requer mais casos de teste é o *entab.c* e o que requer mais associações é o *find.c* e não o *getcmd.c* como é para os outros critérios.

Outro ponto a ser considerado é o número de elementos não executáveis. Eles podem aumentar o esforço e o custo de testar porque isto é determinado manualmente. Os critérios ARCS e BOR não requerem elementos não executáveis. PU e BRO requisitaram a mesma porcentagem de elementos não executáveis, cerca de 20%.

4.2.2 – Strength

Para analisar o strength, utilizam-se as Tabelas de 4.6 a 4.13. Os conjuntos TBOR e TBRO sempre alcançaram 100% de cobertura para o critério ARCS. Este fato mostra que tanto TBOR como TBRO são conjuntos ARCS adequados para todos os programas no experimento. Isto representa que o critério ARCS é o mais fácil de ser satisfeito. Os conjuntos TARCS obtiveram de coberturas relativamente altas para o critério BOR; somente 03 restrições BOR não foram cobertas, mas o mesmo não ocorreu para o critério BRO; 20 restrições executáveis não foram cobertas (24% delas). Isto mostra que é mais fácil satisfazer o critério BOR do que o BRO. Com respeito ao TPU, observou-se que as coberturas BOR e BRO não foram 100%. Não existe diferença entre a cobertura dos conjuntos TPU e TARCS para BOR. Mas a cobertura do critério BRO alcançada por TPU, é maior que a cobertura do conjunto TARCS. Pode-se concluir com isto, que é mais fácil satisfazer BRO dado que PU foi satisfeito do que dado que ARCS foi satisfeito. TBOR e TBRO cobriram, respectivamente, 64,1% e 77,2% das associações executáveis. Nem PU incluiu BOR e BRO como também BOR e BRO não incluíram PU. Eles são incomparáveis. Entretanto, pode-se observar que é mais fácil satisfazer PU dado que BRO foi satisfeito do que dado ARCS ou BOR forem satisfeitos. É mais difícil satisfazer PU do que os outros critérios.

Estes resultados mostram uma relação empírica entre os critérios estudados. Esta relação, baseada no strength, pode ser considerada para propor uma estratégia de aplicação dos critérios, ou seja, para estabelecer uma ordem para aplicá-los, e pode ser vista na Figura 4.1:

ARCS → BOR → BRO → PU

Figura 4.1 – Estratégia de Aplicação dos Critérios do Experimento.

5 – Conclusão

Esse trabalho descreveu a ferramenta PredTOOL, que apóia a utilização dos critérios BOR e BRO para programas escritos em C. A ferramenta contribui com a atividade de teste, melhorando a confiabilidade dos programas testados nessa linguagem, amplamente utilizada. Ela permite que uma estratégia de teste, que envolva a aplicação de diferentes e complementares critérios seja utilizada, pois a aplicação manual desses critérios é impraticável.

Uma outra contribuição da PredTOOL é a condução de experimentos. Apresentou-se um experimento que mostra resultados de uma avaliação empírica de critérios de teste baseados em predicados, envolvendo os critérios BOR e BRO e outros critérios estruturais: Todos-Arcos e Todos Potenciais-Usos. As ferramentas PredTOOL e POKE-TOOL foram utilizadas no experimento.

Os resultados foram analisados com relação a dois fatores: custo e strength. Observa-se que o critério PU é o mais custoso e o mais difícil de satisfazer. Os critérios BOR e ARCS são os mais práticos e menos custosos, além de não apresentarem elementos não executáveis. No entanto, deve-se considerar que esses casos de teste exigidos a mais pelo critério PU podem implicar em um maior número de defeitos revelados. Portanto, a ordem proposta na seção anterior deve considerar também os módulos mais críticos e a confiabilidade requerida para o sistema. O relacionamento entre os critérios obtido neste experimento foi utilizado para estabelecer uma estratégia de aplicação destes critérios. Por exemplo, poder-se-ia aplicar primeiramente um critério mais fraco para obter o conjunto de teste inicial T e após isto gerar os casos de teste adicionais para cobrir os outros critérios seguindo a ordem de aplicação

proposta. Na maioria dos casos, não se aplicam todos os critérios devido aos altos custos. O critério PU é o mais forte, e muito caro para ser utilizado, poderia ser aplicado somente em módulos críticos do sistema ou em softwares onde a confiabilidade requerida é bastante alta. Esta estratégia tal como aqui descrita pode reduzir os esforços e o custo de teste.

Outros experimentos, comparando-se os critérios BOR e BRO com critérios baseados em erros poderão ser realizados. Outros fatores de comparação, como a eficácia e outro conjunto de programas poderão ser considerados.

A ferramenta PredTOOL também pode ser modificada. Pretende-se construir uma interface gráfica para tornar seu uso mais fácil e implementar mecanismos que auxiliem o usuário na determinação de restrições não executáveis.

Referências

[1] CHAIM, M. L. *POKE-TOOL - uma ferramenta para suporte ao teste baseados em fluxo de dados.* Tese de Mestrado, DCA/FEEC/Unicamp, Campinas – São Paulo. 1991.

[2] CHAIM, M. L.; MALDONADO, J.C.; VERGILIO, S. R. *Critérios potenciais usos: análise de aplicação de um benchmark.* VI Simpósio Engenharia de Software, pg 357-371. Gramado, Brasil, 1992.

[3] Frankl F. G. Weyuker E. J. *Data flow testing tools*. In Softfair II, pages 46–53, San Francisco, December 1985.

[4] GRAHAM, D. R. Testing. In: *Encyclopedia of software engineering*. J. Wiley,. v. 2, p. 1330-1353. 1994.

[5] HORGAN, J. R.; LONDON, S. *ATAC- Automatic Test Coverage Analysis for C Programs.* Bellcore Internal Memorandum. June - 1990.

[6] KERNINGHAN, B. W.; PLAUGER, E. J. *Software tools in pascal***.** Addison-Wesley Publishing Company Reading. Massachusetts, USA. 1981.

[7] MALDONADO, J. C.; *Critérios Potenciais-Usos: Uma Contribuição ao Teste Estrutural de Software;* Tese de Doutorado, DCA/FEE/UNICAMP - Campinas, SP, Brasil, 1991.

[8] MALDONADO, J. C.; VINCENZI, A. M. R.; BARBOSA, E. F.; SOUZA, S. R. S. *Aspectos teóricos e empíricos de teste de cobertura de software.* In: ESCOLA DE INFORMÁTICA DA SBC DA REGIÃO SUL, 6., maio 1998. Anais... Blumenau: SBC, 1998.

[9] MYERS, G.; The Art of Software Testing, Wiley, 1979.

[10] PARDKAR, A.; TAI, K. C.; VOUK, M. A. *Empirical studies of precate-basead software testing.* In IEEE Sump. Software Reliablility. Pages 55-65. 1994.

[11] PRESSMAN, R. S.; *Engenharia de Software*. Tradução de Jose Carlos Barbosa; revisão técnica José Carlos Maldonado, Paulo César Masiero, Rosely Sanches. Editora Makron Books, 1995.

[12] PRESSMAN, R. S**.** *Software engineering: a practitioner's approach*. New York:McGraw-Hill, 1992.

[13] RAPPS, S.; WEYUKER, E. J.; *Data Flow Analysis Techniques for Test Data Selection*, Proceedings of International Conference on Software Engineering, páginas 272-278, 1982.

[14] RAPPS, S.; WEYUKER, E. J.; *Selecting Software Test Data using Data Flow Information,* IEEE Transactions on Software Engineering, SE-11(4), Abril, 1985.

[15] TAl, K. C.; *Predicate-based test Generation for computer programs.* Proceedings of International Conference on Software Engineering, páginas 267 -276, IEEE Press. 1993.

[16] VERGILIO, S. R.; MALDONADO, J. C.; JINO, M. *Constraint based criteria: An approach for test case selection in the structural test***.** Journal of Eletronic Testing. Vol 17(2): 175-183. April 2001.

[17] VERGILIO, S. R.; MALDONADO, J. C.; JINO, M. *Caminhos não executáveis na automação das atividades de teste***.** In: Simpósio Brasileiro De Engenharia De Software. p. 343-356. nov. 1992

[18] WEYUKER, E. J. *An empirical study of the complexity of data flow testing*. In Proceedings Of The Second Workshop On Software Testing, Verification E Validation, pages 188–195, Computer Science Press, Banff-Canada, July, 1988.

[19] WEYUKER, E. J. *The cost of data flow testing: an empirical study***.** IEEE Transactions On Software Testing, Verification, Validation and Analysis. pages Vol. SE-16(2)121-128, February 1990.

[20] WEYUKER, E. J. WEISS, S. N. HAMLET. R. G. *Comparison of program testing strategies*. In 4th Symposium on Software Testing, Analysis and Verification, pages 154–164, Victoria, British Columbia, Canadá, 1991. ACM Press.

[21] WONG, A. P.; MALDONADO, J. C. *Mutation versus all-uses: an empirical evaluation of cost, strength e effectivenes.*. In Software Quality and Productivity – Theory, Practice, Education and Trainning. Hong-Kong, December, 1994.

Un algoritmo GRASP para resolver el problema de la programación de tareas dependientes en máquinas diferentes (task scheduling)

Manuel Tupia
Pontificia Universidad Católica del Perú, Departamento de Ingeniería
Av. Universitaria cuadra 18 S/N
Lima, Perú, Lima 32
tupia.mf@pucp.edu.pe

David Mauricio
UPG-FISI Universidad Nacional Mayor de San Marcos
Av. Germán Amézaga S/N.
Lima, Perú, Lima 1
dms@terra.com

Abstract

The industrial planning has experienced notable advances from his origins around the middle of the century XX not only a meal in the efficiency in application importance inside all the industries where it is used, and sophistication of the algorithms that try to resolve the problems that generate all its existent variants. The interest for the heuristic-methods application in front of to give answers to the problems of the area of planning need has taken us to develop new algorithms to resolve one of the problem variants of the planning from the Artificial Intelligence's point of view: The programming of tasks or task scheduling: once an tasks set was given to be programmed in determined machines group, finding an order once was made suitable of execution that minimize the time once was accumulated total of processing of the machines or makespan. The present work GRASP to resolve programming the problem of dependent tasks in different machines shows a heuristic goal.

Keywords: GRASP algorithms, task scheduling, combinatorial optimization, meta heuristics, Artificial Intelligence

Resumen

La planificación industrial ha experimentado notables avances desde sus orígenes a mediados del siglo XX tanto en importancia de aplicación dentro de todas las industrias en donde es usada, como en la eficiencia y sofisticación de los algoritmos que buscan resolver los problemas que generan todas sus variantes existentes. El interés por la aplicación de métodos heurísticos ante la necesidad de dar respuestas a los problemas del área de planificación nos ha llevado a desarrollar nuevos algoritmos para resolver una de las variantes del problema de la planificación desde el punto de vista de la Inteligencia Artificial: la programación de tareas o task scheduling: dado un conjunto de tareas dependientes de una línea de producción a ser programadas en un determinado grupo de máquinas diferentes, encontrar un orden adecuado de ejecución que minimice el tiempo total de trabajo de las máquinas o makespan. El presente trabajo muestra una meta heurística GRASP para resolver dicha variante del problema del task scheduling.

Palabras claves: algoritmos GRASP, programación de tareas, optimización combinatoria, meta heurísticas, Inteligencia Artificial

1. INTRODUCCIÓN

El problema de la programación de tareas o task scheduling presenta sus antecedentes en la planificación industrial [1] y en la programación de trabajos de procesadores en los inicios de la micro electrónica [2]. Ese tipo de problema puede definirse, desde el punto de vista de la optimización combinatoria [3], como sigue:

- Se tienen M máquinas, también denominadas procesadores.
- Se tienen N tareas cada una de ellas con duración T_{ij} unidades de tiempo (el tiempo que demora en ser ejecutada la tarea j en la máquina i).
- El objetivo es programar las N tareas en las M máquinas procurando el orden de ejecución más apropiado, cumpliendo determinadas condiciones que satisfagan la optimalidad de la solución requerida para el problema [4].

El problema presenta una serie de variantes dependiendo de la naturaleza y el comportamiento tanto de las tareas y de las máquinas. Una de las variantes más difíciles de plantear, debido a su alta complejidad computacional es aquella en donde ***las tareas son dependientes y las máquinas diferentes***.

En esta variante cada tarea presenta una lista de tareas que la preceden y para ser ejecutada deben esperar el procesamiento de toda la lista completa de predecesoras. A esta situación hay que agregarle la característica de heterogeneidad de las máquinas: cada una de ellas se demora en ejecutar una tarea tiempos distintos entre sí. El objetivo será minimizar el tiempo acumulado de ejecución de las máquinas, conocido en la literatura como ***makespan***.

Al observar el estado del arte del problema vemos que tanto su aplicación práctica de forma directa en la industria y su importancia académica (al ser un problema NP-difícil) se justifica el diseño de un algoritmo heurístico que busque una solución óptima al problema, dado que no existen métodos exactos para resolver el problema. En muchas industrias como las del ensamblado, embotellado, manufactura, etc., vemos líneas de producción en donde los períodos de espera por trabajo de las máquinas involucradas y el ahorro del recurso tiempo son temas muy importantes y requieren de una conveniente planificación.

A partir de la definición dada, podemos presentar el problema como un modelo matemático de optimización combinatoria en su forma más general, en la siguiente figura:

Minimizar X_0

s. a. $$X_0 \geq \sum_{i=1}^{n} T_{ij} * X_{ij} \quad \forall \ j \in 1..M$$

$$X_0 \geq \sum_{j=1}^{m} X_{ij} = 1 \quad \forall \ i \in 1..N$$

Figura 1: Modelo matemático del task scheduling

El presente escrito propone una heurística del tipo goloso-miope o voraz para resolver la variante antes mencionada del task scheduling.

2. ALGORITMOS META HEURÍSTICOS

Los procedimientos meta heurísticos son una clase de métodos aproximados que están diseñados para resolver problemas difíciles de optimización combinatoria, en los que las soluciones heurísticas clásicas no son efectivos. Proporcionan un marco general para crear nuevos algoritmos híbridos combinando diferentes conceptos derivados de la Inteligencia Artificial (en adelante IA), la evolución biológica y los mecanismos estadísticos.

Permiten generalizar los procedimientos de búsqueda heurística, generando esquemas que pueden ser adaptados a diversos dominios de problemas. Amplían los criterios y modos de optimización de funciones de las heurísticas reduciendo el riesgo de estancamiento en óptimos locales.

Se sitúan conceptualmente por encima de los heurísticos en el sentido que guían el diseño de éstos: así, al enfrentarnos a un problema de optimización, podemos escoger cualquiera de estos métodos para diseñar un algoritmo específico que lo resuelva aproximadamente. Pasaremos a describir una técnica meta heurística en particular: las llamadas técnicas GRASP

2.1 Algoritmos GRASP

Una técnica meta heurística es la de los algoritmos ***GRASP*** (del inglés ***Greedy Randomized Adaptative Search Procedure***). Esta técnica fue desarrollada por T. Feo y M. Resende a finales de los años 80 [10]. Mientras que el criterio goloso permitía seleccionar solamente el mejor valor de la función objetivo c a tratar, los algoritmos GRASP relajan o amplían este criterio de tal manera que, en vez de seleccionar un único elemento, forma un conjunto de elementos candidatos a ser parte del conjunto solución y que cumplen ciertas condiciones; es sobre este conjunto formado, que realizará una selección aleatoria de algún elemento. Tienen esta denominación por lo siguiente:

- Son procedimientos de búsqueda: dentro de un espacio de posibles soluciones se realizan búsquedas sin evaluar a todos los elementos del problema.
- Son voraces: porque en cada evaluación escoge a los mejores candidatos que cumplan ciertas condiciones.
- Son adaptativas: porque se adapta a la estructura de la instancia del problema que pretende resolver.

Son aleatorias: porque de entre las candidatas escoge aproximadamente al azar, aquellas que finalmente formarán parte de la solución

Procedimiento GRASP (Instancia del problema)
1. Leer (Instancia)
2. Mientras <no se cumpla condición de parada> hacer
2.1 Fase de Construcción (S_k)
2.2 Fase de mejoría (S_k)
Fin Mientras
3. Retornar (Mejor S_k)
Fin GRASP

Sobre este algoritmo podemos afirmar:

- Línea 1: se ingresan los datos que conforman la instancia del problema
- Línea 2: el proceso GRASP seguirá mientras no se cumpla una condición de parada. La condición de parada puede ser de varios tipos: optimalidad (el resultado obtenido presenta cierto grado de aproximación a la solución exacta o es lo suficientemente óptimo); número de ejecuciones realizadas (cantidad de iteraciones); tiempo de procesamiento (el algoritmo será ejecutado durante un determinado período de tiempo)
    - Líneas 2.1 y 2.2: se ejecutan las dos fases principales de un algoritmo GRASP, a seguir: ***etapa de construcción de la solución golosa adaptada y aleatoria; y la etapa de mejoría de la solución construida anteriormente.***
- Línea 3: finalmente se devuelve el resultado de la aplicación del algoritmo.

2.2 Fase de construcción GRASP

Esta etapa consiste en construir una solución golosa relajada o ampliada en donde exista más de una candidata a ser solución y a partir de la cual, se seleccionará aleatoriamente elementos para formar la solución final. Recordemos que el criterio goloso que buscaba optimizar la función objetivo podía ser planteado de la siguiente forma: $Mejor\{c(x) : x \in N\}$ en donde c es una función golosa.

Aquí el criterio GRASP modifica al goloso ampliándolo, de forma tal que se pueda construir una lista de varios elementos candidatos a ser solución. Primero determina los mejores y valores de c para luego formar la lista de candidatas como vemos en la siguiente figura:

$$\beta = Mejor\{c(x) : x \in N\}$$
$$\tau = Peor\{c(x) : x \in N\}$$

Figura 2: Valores extremos de la función c

El conjunto de candidatas RCL (del inglés: *Restricted Candidates List*): se formará con todos los elementos x de N que cumplan con la condición: $RCL = \{x \in N : \beta \leq c(x) \leq \beta + \alpha(\tau - \beta)\}$

Al parámetro α que aparece en la definición de RCL se le conoce como ***parámetro de relajación*** y es el responsable de tornar la solución que surgirá, más amplia que la solución voraz. Del conjunto RCL se toma un elemento al azar que pasa a formar parte del conjunto S, repitiéndose el proceso hasta que se hayan procesado todos los elementos de N.

Una estrategia aleatoria viene a ser particularmente útil cuando existen varias formas para realizar un paso dentro de un algoritmo en general, y donde resulta difícil garantizar totalmente que alguna de ellas es la mejor elección. Eso ocurre con el criterio goloso al no analizar más allá los efectos de la selección del mejor elemento de c en cada iteración.

El criterio GRASP indica que el candidato a ser parte del conjunto solución es así, un elemento seleccionado en forma aleatoria desde el conjunto RCL (no pertenece a S y su valor de la función objetivo no necesariamente es el mejor en ese instante).Como se puede apreciar, depende del valor de α empleado.

Según el valor que tome **α,** el comportamiento del criterio GRASP puede variar desde meramente aleatorio hasta volverse totalmente goloso. Veamos:

$\alpha = 0$	→	Criterio totalmente aleatorio
$\alpha = 1$	→	Criterio totalmente goloso
$0 < \alpha < 1$	→	Criterio GRASP convencional que requerirá de calibración adecuada a cada problema.

Figura 3: Posibles valores de la constante de relajación α

3. MÉTODOS EXISTENTES PARA RESOLVER EL PROBLEMA DE LA PROGRAMACIÓN DE TAREAS Y SUS VARIANTES

Resumiremos aquí partes importantes de trabajos que pretenden resolver el problema tanto de forma exacta como aproximada.

3.1 Métodos Existentes

Las soluciones existentes que pretenden resolver el problema de la planificación industrial en general pueden ser divididos en dos: métodos exactos y métodos aproximados. Los métodos exactos pretenden hallar un plan jerárquico único analizando todos los posibles ordenamientos de las tareas o procesos involucrados en la línea. Sin embargo, una estrategia de búsqueda y ordenamiento que analice todas las combinaciones posibles es computacionalmente cara y sólo funciona para algunos tipos de instancia.

Los métodos aproximados por su parte sí buscan resolver las variantes más complejas en las que interviene el comportamiento de tareas y máquinas como se mencionó en el apartado anterior. Ellos no analizan exhaustivamente todas las posibles combinaciones de patrones del problema, sino que más bien eligen los que cumplan determinados criterios. Obtienen finalmente, soluciones lo suficientemente óptimas para las instancias que resuelven, lo que justifica su uso.

Los sistemas existentes en el mercado que pueden resolver el problema, no aplican técnicas heurísticas o meta heurísticas para su desarrollo [6]. Este hecho se puede deber a la antigüedad de los mismos y los métodos empleados (exactos) [7][8]. Al no conseguirse soluciones exactas para muchos de los ámbitos industriales aplicados se obliga al uso de técnicas híbridas o de otro tipo que se ocupen de arrojar resultados aproximados y óptimos.

3.1.1. Job Scheduling

Dentro de la teoría de colas podemos encontrar la primera y más estudiada variante del scheduling: el problema de la planificación de trabajos en cola o atención en cola conocido en la literatura como job scheduling [9]. Este problema consiste en:

- Dado un lote finito de trabajos a ser procesados
- Dado un infinito número de procesadores (máquinas).
- Cada trabajo (job) esta caracterizado por un conjunto de operaciones convenientemente ordenadas; por su parte, las máquinas pueden procesar una única tarea a la vez, sin interrupciones.

El objetivo del JSP es encontrar una programación determinada que minimice el tiempo de procesamiento.

Existen numerosos algoritmos heurísticos planteados para resolver el problema del job scheduling, entre los que destacamos:

- Usando algoritmos GRASP: Binato, Hery y Resende [11]
- Usando algoritmos Genéticos: trabajos de Goncalves, De Magalahes y Resende [12] y Davis [13]
- Usando algoritmos de Ramificación y Acotación: trabajo de Brucker, Jurisch y Sievers [14]

3.1.2. Task Scheduling

Los algoritmos para esta variante pueden ser usados para resolver complicados problemas de planificación o programación de acciones y operaciones en células de trabajo. Se plantea un determinado número finito de máquinas y tareas y se busca, al igual que en el JSP, una programación adecuada donde se minimice el tiempo de procesamiento de lote o makespan.

Por la naturaleza de las máquinas y las tareas, puede hacerse la siguiente subdivisión vista anteriormente:

- Máquinas idénticas y tareas independientes

- Máquinas idénticas y tareas dependientes
- Máquinas diferentes y tareas independientes
- Máquinas diferentes y tareas dependientes: el modelo más complejo que será motivo de estudio en este trabajo.

Algunos algoritmos planteados son:
- Usando algoritmos voraces: Campello, Maculan [3] para máquinas idénticas.
- Usando algoritmos voraces: Tupia [15] para máquinas diferentes y tareas independientes
- Usando algoritmos GRASP: Tupia [15] para máquinas diferentes y tareas independientes

4. ALGORITMO GRASP PROPUESTO

Debemos partir del supuesto que se tiene una instancia de trabajo completa que incluya lo siguiente: cantidad de tareas y máquinas (N y M respectivamente); matriz de tiempos de ejecución T y lista de predecesoras por tarea.

4.1 Estructuras de Datos Usadas por el Algoritmo

Consideremos que en el lote hay al menos una tarea sin predecesoras que se convertirá en la inicial, así como no existen referencias circulares entre las predecesoras de las tareas que impidan su correcta programación. En la siguiente figura vemos las estructuras de datos que vamos a necesitar para la presentación del algoritmo:

N: número de tareas $J_1, J_2, ..., J_N$

M: número de máquinas $M_1, M_2, ..., M_M$

Matriz T: $[T_{ij}]_{MxN}$ **de tiempos de procesamiento, donde cada entrada representa el tiempo que se demora la máquina j-ésima en ejecutar la tarea i-ésima.**

Vector A: $[A_i]$ **de tiempos de procesamiento acumulado, donde cada entrada** A_i **es el tiempo de trabajo acumulado de la máquina** M_i

P_k **: Conjunto de tareas predecesoras de la tarea** J_k

Vector U: $[U_k]$ **de tiempos de finalización de cada tarea** J_k

Vector V: $[V_k]$ **de tiempos de finalización de las tareas predecesoras de** J_k**, donde se cumple que** $V_k = \max\{U_r\}, J_r \in P_k$

S_i **: Conjunto de tareas asignadas a la máquina** M_i

E: Conjunto de tareas programadas

C: Conjunto de tareas candidatas a ser programadas

Figura 4: Estructuras de datos usadas por el algoritmo

4.2 Consideraciones Generales

Vamos a plantear dos criterios de selección durante el desarrollo del algoritmo GRASP lo que nos va a llevar a generar dos constantes de relajación en vez de una sola:
- Un criterio GRASP de selección aleatorio de la mejor tarea a programarse, empleando la constante de relajación α.
- Un criterio de selección aleatorio de la mejor máquina que ejecutará la tarea seleccionada antes, empleando un parámetro θ adicional.

4.2.1 Criterio de Selección de la Mejor Tarea

Este criterio se basa en los mismos principios que el algoritmo voraz presentado antes:
- Identificar las tareas aptas para ser programadas: es decir, las que no han sido programadas aún y sus predecesoras ya han sido ejecutadas (o no presentan predecesoras).
- Para cada una de las tareas aptas, generar la misma lista que en el algoritmo goloso: establecer los tiempos de ejecución acumulado a partir de la finalización de la última tarea predecesora que se ejecutó.

- Obtener el menor elemento de cada lista y almacenarlos en otra lista de mínimos locales. De esta nueva lista de mínimos locales seleccionaremos: el máximo y el mínimo valor las variables: *peor y mejor* respectivamente.
- Formaremos la lista de tareas candidatas RCL analizando cada entrada de la lista de mínimos locales: si la entrada correspondiente cumple con estar dentro del intervalo **[mejor, mejor + α* (peor-mejor)],** entonces pasa a formar parte de RCL.
- Se escoge una tarea al azar de las que constituyen RCL.

4.2.2 Criterio Goloso de Selección de la Mejor Máquina
Una vez seleccionada una tarea desde RCL, procedemos a buscar la mejor máquina que la pueda ejecutar. Los pasos a seguir son los siguientes:

- Se forma nuevamente el vector de tiempos acumulados en función de las predecesoras de la tarea j-ésima que está siendo objeto de análisis.
- Se encuentran los valores máximos y mínimos de en las variables: *peor y mejor* respectivamente.
- Procedemos a formar la lista de máquinas candidatas ***MCL***: toda máquina que ejecute a la tarea j-ésima en un tiempo que se encuentre en el intervalo **[mejor, mejor + θ* (peor-mejor)]** forma parte de MCL.
- Igualmente seleccionamos al azar una de las máquinas, que será la ejecutora de la tarea j-ésima.

4.3 Presentación del algoritmo

Inicio AlgoritmoGRASP_Construccion (M, N, T, A, α, θ)

1. Leer N, M, α, θ, $J_1, J_2, ..., J_N$

2. Para i:1 a N, hacer

Para j: 1 a M, hacer

Leer T_{ij}

3. Para i:1 a M, hacer

Inicio

A_i **= 0**

$S_i = \phi$

Fin Para

4. Para k: 1 a N, hacer

Inicio

U_k **= 0**

V_k **= 0**

Fin Para

5. E $= \phi$

6. Mientras $|E| \neq N$ **hacer**

Inicio

6.1 C $= \phi$

6.2 mejor $= +\infty$

6.3 peor = 0

6.4 Para ℓ: 1 a N, hacer

Si ($P_\ell \subseteq E$) $\wedge$ ($J_\ell \notin E$) $\Rightarrow$ C = C $\cup$ { J_ℓ }

6.5 $B_{min} = \phi$

6.6 Para cada $J_\ell \in$ **C hacer**

Inicio

6.6.1 $V_l = \max_{J_l \in P_k} \{U_l\}$

6.6.2 $B_{min} = B_{min} \cup Min_{p \in [1,M]} \{T_{pl} + \max\{A_p, V_l\}\}$

Fin Para

{Selección de la mejor tarea. Formación de RCL}

6.7 mejor = $Min\{B_{\min}\}$

6.8 peor = $Max\{B_{\min}\}$

6.9 $RCL = \phi$

6.10 Para cada $J_\ell \in$ **C hacer**

Si $Min_{p \in [1,M]} \{T_{pl} + \max\{A_p, V_l\}\} \in$ **[mejor, mejor + α* (peor-mejor)]** $\Rightarrow$

RCL = RCL $\cup$ **{** J_ℓ **}**

6.11 k = $ArgAleatorio_{J_l \in RCL} \{RCL\}$

{Selección de la mejor máquina}

6.12 $MCL = \phi$

6.13 mejor = $Min_{p \in [1,M]} \{T_{pk} + \max\{A_p, V_k\}\}$

6.14 peor $= Max_{p \in [1,M]} \{T_{pk} + \max\{A_p, V_k\}\}$

6.15 Para i: 1 a M hacer

Si $T_{ik} + \max\{A_i, V_k\} \in$ **[mejor, mejor + θ* (peor-mejor)]** $\Rightarrow$

MCL = MCL $\cup \{M_i\}$

6.16 i = $ArgAleatorio_{M_i \in MCL} \{MCL\}$

6.17 $S_i = S_i \cup \{J_k\}$

6.18 E = E $\cup \{J_k\}$

6.19 $A_i = T_{ik} + \max\{A_i, V_k\}$

6.20 $U_k = A_i$

Fin Mientras

7. makespan = $\max_{k \to 1..M} A_k$

8. Retornar makespan, $S_i \ \forall i \in [1, M]$

Fin AlgoritmoGRASP_Construcción

Figura 5: Algoritmo GRASP Construcción.

4.3.1. Comentarios sobre el algoritmo GRASP propuesto

- Líneas 1-5: ingreso de las variables e inicialización de las estructuras de datos necesarias para la instancia de trabajo: N, M, T, A, U, V, E, S_i para cada máquina, α, θ, etc.
- Línea 6: al igual que en el algoritmo voraz, el proceso acaba cuando en el conjunto E se encuentren todas las tareas (E conjunto de las tareas programadas)
    - Línea 6.1: se inicializa en vacío la lista de tareas aptas C
    - Línea 6.2: se inicializan las variables mejor y peor con las que se trabajarán los intervalos de los criterios de relajación.
    - Línea 6.4: se forma la lista de tareas aptas C
    - Línea 6.5: se inicializa en vacío la lista de mínimos locales B_{min}
    - Líneas 6.6 – 6.6.2: se actualiza las entradas correspondientes de la lista V; se forma la lista B_{min} agregándole cada menor elemento de la lista de tiempos acumulados de cada tarea.
    - Líneas 6.7-6.8: se le asignan los valores máximos y mínimos de B_{min} a las variables peor y mejor respectivamente.
    - Línea 6.9: se inicializa en vacío la lista RCL
    - Líneas 6.10-6.11: se forma la lista RCL cuando se cumpla la condición $Min_{p\in[1,M]}\{T_{pl} + \max\{A_p, V_l\}\} \in$ [mejor, mejor + α* (peor-mejor)]; luego se escoge un elemento al azar de ésta lista (k)
    - Líneas 6.12-6.16: se escoge el mínimo y el máximo tiempo de ejecución para la tarea escogida en 6.11. Se formará MCL a partir de las máquinas que ejecuten dicha tarea cumpliendo la condición: $T_{ik} + \max\{A_i, V_k\} \in$ [mejor, mejor + θ* (peor-mejor)]. Finalmente también se escoge una máquina de forma aleatoria (i)
    - Líneas 6.17-6.20: se actualizan las estructuras E, A, U y S_i donde corresponda.
- Línea 7: se determina el makespan como la mayor entrada de A
- Línea 8: se devuelven los resultados de asignación hallados.

5. EXPERIENCIAS NUMÉRICAS

Las instancias de prueba con las que se probó el algoritmo están conformados por la cantidad de máquinas, tareas y por una matriz de tiempos de ejecución. Los valores que se manejarán serán los siguientes:

- Número de tareas N: en el intervalo 100…250 tomando como puntos de referencia los valores de 100, 150, 200, 250.
- Número de máquinas M: un máximo de 50 máquinas tomando como puntos de referencia los valores de 12, 25, 37, 50
- Matriz de tiempos de proceso: se generará de forma aleatoria con valores entre 1 y 100 unidades de tiempo.[1]

En total tenemos 16 combinaciones para las combinaciones ***máquinas-tareas***. De la misma forma, para cada combinación se generarán 10 instancias distintas, lo que arroja un total de ***160 problemas test*** realizados**.** Ante la no existencia en la literatura de instancias de prueba predeterminados para el problema en cuestión, decidimos enfrentar los resultados con los que ofrecía un ***algoritmo goloso o voraz*** [5], diseñado por los mismos autores. ***Igualmente*** se procedió a crear instancias lo suficientemente pequeñas (manejables) para probarlas y obtener su solución exacta planteándolas como problemas de la programación lineal, de la mano del modelo matemático presentado en anteriormente.

Así pues, creamos instancias que iban a ser resueltas por medio del programa **LINDO** [16], para obtener sus soluciones exactas. Luego, compararíamos esas soluciones exactas con los resultados arrojados por el algoritmo propuesto. Las instancias usadas para ser resueltas de forma exacta han tenido los siguientes tamaños (recuérdese que N es el número de tareas y M el número de máquinas):

[1] : Notemos que un tiempo de ejecución muy alto (+∞) puede interpretarse como que la máquina no ejecuta una tarea determinada. Usar en este caso un tiempo de ejecución igual a 0 podría confundirse como que la máquina ejecuta *tan rápido la tarea* que se puede asumir que lo hace de forma instantánea, sin ocupar tiempo.

N	M
6	3
8	3
12	3
15	3
10	5
15	5
20	5
25	5

Tabla 1: Tamaños de las instancias de prueba exacta

Máquinas\ Tareas	GOLOSO	GRASP C			EFICIENCIA
	Makespan	Makespan	α	θ	%
100 \ 12	213.3	193.7	0.26	0.04	9.19%
100 \ 25	113.7	108.6	0.24	0.01	4.49%
100 \ 37	76.5	74.3	0.155	0.01	2.88%
100 \ 50	59.1	57.8	0.145	0.01	2.20%
150 \ 12	283.1	258.2	0.25	0.01	8.80%
150 \ 25	125.2	114	0.2055	0.01	8.95%
150 \ 37	86.1	84.1	0.155	0.01	2.32%
150 \ 50	68.6	67.4	0.115	0.01	1.75%
200 \ 12	325.9	307	0.07	0.01	5.80%
200 \ 25	127	117	0.247	0.01	7.87%
200 \ 37	109.8	105.2	0.112	0.01	4.19%
200 \ 50	76.4	74.6	0.155	0.01	2.36%
250 \ 12	411.8	377.1	0.15	0.01	8.43%
250 \ 25	183.5	168.2	0.28	0.01	8.34%
250 \ 37	125.3	119.6	0.26	0.01	4.55%
250 \ 50	96.5	91.1	0.123	0.01	5.60%
Eficiencia algoritmo GRASP sobre el algoritmo Goloso					**5.48%**

Tabla 2: Eficiencia entre algoritmo GRASP y goloso para instancias grandes

Matriz	N	M	Resultado del LINDO	Algoritmo voraz	Algoritmo GRASP	% Exacto/Voraz	% Exacto/GRASP	a	θ
6x3_0	6	3	41	41	41	0.00%	0.00%	0.01	0.01
6x3_1	6	3	109	109	109	0.00%	0.00%	0.01	0.01
6x3_2	6	3	98	108	98	10.20%	9.26%	0.01	0.01
8x3_0	8	3	84	83	83	1.19%	0.00%	0.01	0.01
8x3_1	8	3	153	180	153	17.65%	15.00%	0.01	0.01
8x3_2	8	3	94	94	94	0.00%	0.00%	0.01	0.01
12x3_0	12	3	252	252	252	0.00%	0.00%	0.01	0.01
12x3_1	12	3	238	253	238	6.30%	5.93%	0.01	0.02
12x3_2	12	3	168	168	165	0.00%	1.79%	0.01	0.45
15x3_0	15	3	300	301	262	0.33%	12.96%	0.01	0.41
15x3_1	15	3	193	193	181	0.00%	6.22%	0.47	0.37
15x3_2	15	3	290	292	241	0.69%	17.47%	0.43	0.02
Diferencia						**3.03%**	**5.72%**		

Tabla 3: Eficiencia de la solución golosa para instancias con N igual a 3

Matriz	N	M	Resultado del LINDO	Algoritmo voraz	Algoritmo GRASP	% Exacto/Voraz	% Exacto/GRASP	a	θ
10x5_0	10	5	95	95	95	0.00%	0.00%	0.01	0.01
10x5_1	10	5	106	109	109	2.83%	0.00%	0.01	0.01
10x5_2	10	5	122	123	122	0.82%	0.81%	0.001	0.02
15x5_0	15	5	139	139	119	0.00%	14.39%	0.01	0.09
15x5_1	15	5	157	157	149	0.00%	5.10%	0.06	0.17
15x5_2	15	5	97	103	96	6.19%	6.80%	0.01	0.04
20x5_0	20	5	210	210	189	0.00%	10.00%	0.21	0.01
20x5_1	20	5	132	132	119	0.00%	9.85%	0.01	0.08
20x5_2	20	5	155	156	155	0.65%	0.64%	0.09	0.01
25x5_0	25	5	255	264	255	3.53%	3.41%	0.34	0.07
25x5_1	25	5	249	283	249	13.65%	12.01%	0.37	0.19
25x5_2	25	5	219	219	211	0.00%	3.65%	0.19	0.29
Diferencia						**2.31%**	**5.55%**		

Tabla 4: Eficiencia de la solución golosa para instancias con N igual a 5

6. CONCLUSIONES

Los sistemas existentes en el mercado que pueden ser considerados como planificadores de tareas, no han aplicado técnicas meta heurísticas sino más bien métodos exactos. No buscan la optimización del ordenamiento de las tareas a ejecutar, sino más bien dan énfasis a encontrar planes factibles de ser ejecutados.

Tampoco existen sistemas que planifiquen una línea de producción completa considerando entre otras cosas: el desplazamiento de los productos, la ejecución de las operaciones, la disposición de la maquinaria (layout) y de las facilidades (instalaciones), potenciales reducciones de los tiempos muertos, etc.

Este panorama le imprime un carácter novedoso a la investigación realizada ya que, si bien en la actualidad las técnicas GRASP son frecuentemente empleadas para la resolución de distintas clases de problemas, hasta ahora no se había planteado una GRASP para resolver el problema del task scheduling para tareas dependientes ejecutadas en máquinas diferentes.

Como parte de los nuevos planteamientos presentados en los algoritmos, debemos centrarnos en:

- Un criterio de selección de la mejor tarea a programar
- Un criterio de selección de la mejor máquina a ejecutar la tarea seleccionada con el criterio anterior.

Ambos criterios se amoldaron al tipo de algoritmo presentado: los criterios eran voraces en el caso del algoritmo voraz (una selección inmodificable); o eran adaptativos y aleatorios como en el caso del algoritmo GRASP. En la literatura no existen algoritmos GRASP que consideren un doble criterio de relajamiento al momento de hacer las selecciones correspondientes: las GRASP convencionales, solo formaban una lista RCL de candidatas para las tareas a ser ejecutadas.

Nuestro trabajo amplía esa visión en el caso del algoritmo GRASP, al proporcionarle un componente adaptativo y aleatorio a la selección de la mejor máquina ejecutora; lo que al final afectó a la eficiencia del algoritmo.

- El algoritmo GRASP mejora el 100% de los casos al resultado del algoritmo voraz para las instancias de prueba suficientemente elevadas (proporción 4 a 1, 5 a 1). En instancias pequeñas como mínimo lo iguala al voraz o es superado en un bajísimo porcentaje.
- El porcentaje de mejora del algoritmo GRASP frente al algoritmo goloso es de 5% en promedio.
- El tiempo de ejecución de la fase de construcción GRASP se basa en la cantidad de iteraciones a realizarse: para las instancias mayores (250 x 50) se trataron de realizar más de 1000 iteraciones, pero aparentemente la configuración del hardware de prueba no lo permitió. Esto nos lleva a inferir que, con una configuración de equipo más potente sí se podría pasar esta barrera. El tiempo de CPU empleado no pasaba de los 3 minutos lo que da un promedio también bajo en la duración de la fase de construcción.

La etapa de mejoría GRASP, se limitó a no más que un intercambio de los valores encontrados en la etapa anterior de construcción. ***No brindó mejoras en ninguno de los casos***, lo que nos conduce a pensar que la etapa de construcción es lo suficientemente robusta, para requerir las permutaciones que busquen mejorar su resultado. Es por eso que no se utilizó la mejora.

Referencias

[1]: Miller G., Galanter E. Plans and the Structure of Behavior, Editorial Holt, New York-USA (1960)

[2]: Drozdowski M. Scheduling multiprocessor tasks—an overview, European Journal . Operation Research 94 (1996) 215–230.

[3]: Campello R., Maculan N. Algoritmos e Heurísticas Desenvolvimiento e avaliaçao de performance, Apolo Nacional Editores, Brasil (1992).

[4]: Pinedo M. Scheduling, Theory, Algorithms and Systems, Prentice Hall (1995 y 2002)

[5]: Cormen T.; Leiserson Ch.; Rivest R. Introduction to Algorithms (2da edición), MIT Press, Editorial McGraw Hill, Inglaterra, 2001

[6]: Rauch W. Aplicaciones de la inteligencia Artificial en la actividad empresarial, la ciencia y la industria - tomo II. Editorial Díaz de Santos, Madrid España pp. 685 (1989).

[7]: Kumara P. Artificial Intelligence: Manufacturing theory and practice. Editorial NorthCross Institute of industrial Engineers, USA pp. 686 (1988).

[8]: Blum A., Furst M. Fast Planning through Plan-graph Analysis. Article of memories from 14th International Joint Conference on Artificial Intelligence, pp. 1636-1642. Ediciones Morgan- Kaufmann - USA (1995).

[9]: Suárez R., Bautista J., Mateo M. Secuenciación de tareas de ensamblado con recursos limitados mediante algoritmos de exploración de entornos, www.upc.es/ sol.upc.es/~suarez/pub.html Instituto de Organización y Control de Sistemas Industriales Universidad Politécnica de Cataluña (1999).

[10]: Feo T., Resende M., Greedy Randomized Adaptive Search Procedure Journal of Global Optimization, número 6, pp. 109-133 (1995).

[11]: Binato S., Hery W., Loewenstern D. y Resende M. A GRASP for Job Scheduling. Technical Report N° 00.6.1 AT&T Labs Research (1999-200)

[12]: Gonçalves J., Magalhäes J., Resende M. A Hibrid Genetic algorith for the Jos Shop Scheduling AT&T Labs Research Technical Report TD-5EAL6J (2002).

[13]: Davis L. Job shop scheduling with genetic algorithms. First International Conference on Genetic Algorithms and their Applications, pp. 136-140. Morgan – Kaufmann USA, (1985).

[14]: Brucker P., Jurisch B. and Sievers B. A branch and bound algorithm for the job-shop scheduling problem. Journal of Discrete Applied Mathematics, número 49, pp. 105-127 (1994).

[15]: Tupia, M. Un algoritmo Goloso Adaptativo y Randómico para resolver el problema de la Programación de Tareas independientes en máquinas homogéneas, Tesis presentada para optar por el título de Ingeniero Informático, Pontificia Universidad Católica del Perú (2001).

[16]: Scharage L. 1997. Optimization modeling with LINDO. Duxbury Press. USA.

Integrando diferentes técnicas de Data Mining en procesos de Web Usage Mining

Luca Cernuzzi
Universidad Católica "Nuestra Señora de la Asunción"
Departamento de Ingeniería Electrónica e Informática
Asunción - Paraguay, C.C. 1683, Fax: +595 21 310587
lcernuzz@uca.edu.py

and

María Liz Molas
Universidad Católica "Nuestra Señora de la Asunción"
Departamento de Ingeniería Electrónica e Informática
Asunción - Paraguay, C.C. 1683, Fax: +595 21 310587
lizmolas@telesurf.com.py

Abstract

Web Usage Mining focuses on techniques to search for patterns in the user behaviour when navigating the Web. This work presents a methodological proposal for the integrated application of different Data Mining techniques, in the KDD general process framework, in order to do Web Usage Mining. The methodological proposal is applied to a case study, trying to describe the users' behaviour of a Web portal. Moreover, the study briefly presents the more relevant results obtained during the analysis.
Keywords: KDD, Web Mining, Web Usage Mining, Association Rules, Clustering

Resumen

Web Usage Mining, se basa en las técnicas para buscar patrones en el comportamiento de los usuarios cuando navegan en la Web. En el presente trabajo se presenta una propuesta metodológica para la aplicación integrada de diferentes técnicas de Data Mining, dentro del marco general del proceso de KDD, para realizar Web Usage Mining. Dicha propuesta metodológica es aplicada a un caso de estudio, para intentar describir el comportamiento de los usuarios de un portal Web. También se presentan los resultados más significativos obtenidos durante el análisis.
Palabras claves: KDD, Web Mining, Web Usage Mining, Reglas de Asociación, Clustering

1. Introducción

El crecimiento explosivo de la Web en los años recientes la ha convertido en una gran fuente de datos disponibles on-line. Entre otras características, la Web no provee a sus usuarios páginas con un diseño estándar, es heterogénea en su contenido tanto en relación con la información disponible como a la calidad. En el ambiente Web existe una variedad de datos que pueden ser estructurados, semi estructurados y no estructurados. Además el volumen de datos que se manejan diariamente en los servidores web es muy grande La Web puede ser vista como una colección no estructurada de páginas e hiperlinks, las páginas son accedidas por una gran variedad de personas con diferentes conocimientos e intereses Esto implica una mayor dificultad en inferir conocimiento con respecto a las bases de datos convencionales. Las páginas son generalmente diseñadas con HTML, con una estructura limitada que dificulta el análisis del contenido que realizan las herramientas automáticas. Todos estos hechos denotan la dificultad del usuario para encontrar información relevante en la Web [2,10].

Knowledge Discovery (KDD) y Data Mining son disciplinas de búsqueda que involucran el estudio de técnicas para buscar patrones en grandes colecciones de datos [6].

La aplicación de las técnicas de Data Mining a la Web, llamada Web Data Mining o más sintéticamente Web Mining, es definida como el estudio de las técnicas de Data Mining que automáticamente descubren y extraen información desde la Web [5].

En este contexto, la técnica de WebUsage Mining, intenta descubrir y extraer patrones de uso o comportamiento a partir de datos de la exploración o la navegación (tal como los registros del los archivos log de acceso a los grandes repositorios de datos Web) [3].

El presente trabajo presenta una propuesta metodológica para la realización de Web Usage Mining y presenta un caso de estudio cuyo objetivo es obtener la descripción del comportamiento del usuario durante la visita al sitio web, base importante para la toma de decisiones de diseño del sitio, en términos de su contenido y estructura, también útil para el desarrollo de políticas de Web caching, transmisiones de red y distribución de los datos.

La metodología propuesta para la realización de Web Usage Mining dentro del marco más general de KDD integra dos técnicas de Data Mining (Reglas de Asociación y Clustering).

Como base para el caso de estudio, se utilizaron los archivos log del servidor web de registros de navegaciones para miles de individuos o usuarios en el Portal de Rieder Internet (un Internet Service Provider local).

El resto del trabajo se estructura de la siguiente forma. En la sección 2 se presentan brevemente los conceptos principales para contextualizar y caracterizar las técnicas de Web Mining. En la sección 3 se presenta una propuesta metodológica para la aplicación de dichas técnicas. En la sección 4 una aplicación a un caso de estudio. En la siguiente sección, el análisis de los resultados obtenidos y finalmente se trazan algunas conclusiones y se perfilan posibles trabajos futuros.

2. Web Mining

En los últimos años ha tomado un creciente interés la disciplina de *Knowledge discovery in databases* – KDD, que consiste en el proceso no trivial de identificar patrones validos, novedosos, potencialmente útiles y entendibles, en los datos [6].

El proceso de KDD es interactivo e iterativo, y envuelve numerosos pasos donde muchas decisiones son tomadas por el usuario. La mayoría de los trabajos que se han realizado sobre KDD se centran en el paso de Data Mining. que consiste en la aplicación de algoritmos específicos que, bajo algunas limitaciones aceptables de eficiencia computacional, produce una enumeración particular de patrones [6].

Cabe mencionar que las técnicas de Data Mining han sido usadas para una variedad de tareas o aplicaciones, sin embargo, desde un punto de vista global, Fayyad [6] propone dos categorías de problemas generales, la *predicción* y la *descripción*. La Predicción se basa en algunas variables o campos de la Base de Datos para predecir valores desconocidos o futuros de otras variables de interés. La Descripción, en cambio, se centra en encontrar patrones interpretables por el ser humano, a partir de la descripción de los datos. Para ambos podemos utilizar alguna de las siguientes tareas básicas: clasificación, regresión, clustering, condensación o resumen, modelado de dependencias, análisis de enlaces y análisis secuencial.

Web Data Mining (o simplemente Web Mining) ha sido definida como la aplicación de las técnicas de Data Mining a datos web [2]. En este sentido, en términos amplios Web Mining puede ser definida como el descubrimiento y el análisis de información útil desde la World Wide Web [5].

El objetivo primario del Web Mining es descubrir patrones interesantes en los accesos a varias páginas del espacio web asociado a un servidor particular [12].

El análisis de los datos de acceso del servidor puede proveer información significante y útil que soporten las decisiones de negocios en el comercio electrónico, que ayuden a incrementar el rendimiento, reestructurar el sitio web para mejorar su efectividad y clasificar a los clientes en grupos de interés para dirigir la publicidad o información en general [5,10].

2.1 Web Usage Mining

En la disciplina de Web Mining recubre particular interés la disciplina que se enfoca el análisis de la información de las visitas a distintas páginas Web en orden a extraer patrones de uso, eso es Web Usage Mining.

Cuando los usuarios de la Web interactúan con un sitio, los datos registrados de su comportamiento son típicamente almacenados en los archivos log del servidor web. Estos archivos pueden contener información sobre la experiencia del usuario en el sitio. El tamaño promedio de los archivos puede ser de varios megabytes diarios, por esto son necesarias técnicas y herramientas que faciliten el análisis de su contenido [2].
El análisis de cómo los usuarios acceden a un sitio es critico para determinar la eficacia de las estrategias de marketing y la optimización de la estructura lógica del sitio web. Debido a numerosas características exclusivas del modelo cliente servidor en la Web, incluyendo diferencias entre la topología física de los repositorios y los caminos de acceso de los usuarios, además de la dificultad en identificar a los usuarios en sesiones o transacciones, es necesario desarrollar un nuevo framework para implementar el proceso de minería [5].

3. Propuesta metodológica para Web Usage Mining

El proceso de KDD típicamente involucra una serie de pasos (varios autores reconocen 9 pasos básicos) organizados en un proceso iterativo de 6 etapas. Si bien, en términos generales esto puede resultar interesante y cubre la mayor parte de los posibles métodos para descubrir conocimiento en bases de datos, consideramos que el proceso aplicado al caso de conocimiento inherente a usuarios que navegan en la Web puede ser simplificado. En la Figura 1 se presenta el sistema de Web Usage Mining adaptado al marco más general del proceso de KDD. Además, para el paso inherente a la búsqueda de patrones de comportamiento, es decir, para la aplicación de técnicas de Data Mining, se considera que a diferencia de la mayoría de los enfoque típicamente adoptados de centrarse en una técnica particular, posiblemente la más adecuada al problema en estudio, puede resultar muy útil la integración de distintas técnicas complementarias. Esta misma observación puede ser valida para la etapa de análisis de resultados, en la cual distintas técnicas podrían confirmar los resultados obtenidos o bien ayudar un análisis más detallado en ciertos aspectos abriendo espacio para nuevos patrones.

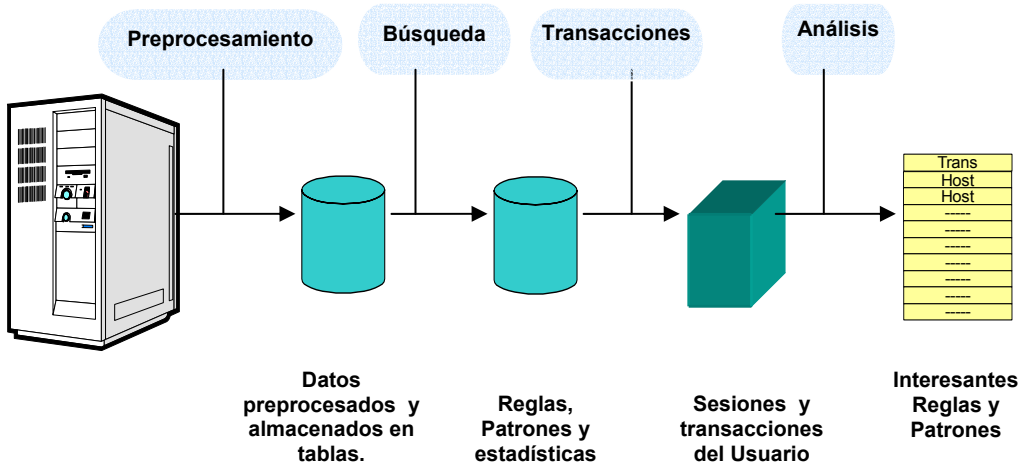


Figura 1 Propuesta metodológica para Web Usage Mining

En el presente trabajo, se presenta la aplicación de la propuesta metodológica a un caso de estudio en el que se modela una colección de sesiones de usuarios a partir de los archivos log de acceso al servidor Web. Para el efecto, se insertan los datos en bruto en tablas relaciónales para facilitar el preprocesamiento y transformándolos en transacciones para luego aplicar en forma integrada dos técnicas estándar de Data Mining ("Reglas de Asociación" y "Clustering"). La aplicación de dos técnicas permite ir validando los resultados de una con la otra y así descubrir nuevos patrones que pueden ser mejor detectables a través de otra técnica. El análisis de los resultados, se realiza mediante la integración de mecanismos de SQL y la técnica de "Visualización" para corroborar los patrones encontrados o descubrir nuevos. Para efectivizar el trabajo de Web Usage Mining, surge la necesidad de contar con un sistema software cuyo objetivo principal sea el preprocesamiento de los accesos log y la identificación de las transacciones de los usuarios, para luego permitir la aplicación de los algoritmos de Data Mining para el descubrimiento de los patrones de comportamiento. También se necesita la creación de una base de datos para almacenar en forma estructurada los accesos log y las transacciones generadas, y como soporte para la aplicación de los algoritmos de Data Mining.
Para el diseño y desarrollo del sistema y de la base de datos y la implementación del proceso de KDD se ha decidido adoptar un enfoque ingenieristico y una metodología orientada a objetos, eso es, el Proceso Unificado Racional [8,9].
Los usuarios que interactúan con el sistema, son el cliente o usuario web quien navega por las páginas del Portal, el Servidor web que registra todos los archivos y páginas solicitados por el usuario web, el analista quien ejecuta los diferentes pasos del Proceso de KDD y el experto que analiza los resultados del Proceso de KDD.

4. Aplicando Técnicas de Web Usage Mining: un caso de estudio

Para poder aplicar y validar las técnicas de Data Mining en general, y en particular aquellas de Web Usage Mining que constituyen el enfoque del presente trabajo, se necesita un gran volumen de datos textuales, y un sitio con una estructura de navegación no trivial. Para el efecto se ha utilizado como caso de estudio el sitio de Rieder Internet (uno de los proveedores de Internet local), y los archivos log de acceso almacenados en su servidor web.

Rieder Internet (http://www.rieder.net.py) ofrece, entre otros servicios, noticias locales e internacionales, asistencia a sus clientes y además soporta otros sub sitios con dominio propio. El sitio cuenta con una variedad de información distribuida en diferentes secciones, tales como mujer, deportes, niños, jóvenes, negocios, tecnología y diversión.

Las técnicas de Data Mining pueden proveer de información cualitativa, en forma de patrones de comportamiento del usuario, para medir la efectividad del sitio, para verificar los supuestos que tiene el contenido de las páginas y para ofrecer un mejor servicio en forma de personalización a sus usuarios.

En este trabajo se ha utilizado un sub conjunto de los archivos log de acceso del servidor web (de abril a julio del 2002), trabajando en forma coordinada con el experto, quien conoce en detalle la estructura del sitio web y los objetivos de la empresa Rieder Internet, en el análisis de los datos y los resultados obtenidos en los diferentes pasos del Proceso de KDD. En la etapa de Data Mining se pretende analizar transacciones de los usuarios para intentar descubrir patrones de comportamiento, es decir describir el comportamiento del usuario cuando navega por el sitio web.

A fin de alcanzar nuestros objetivos hemos seleccionado 2 de las 5 técnicas más conocidas de Data Mining propuestas por distintos autores [5,14]. Las técnicas "Path analysis", que busca los caminos más visitados en el sitio web, y "Sequential patterns**",** que busca las transacciones ordenadas en el tiempo, donde la presencia de un conjunto de elementos es seguida por otro, no están directamente relacionadas con nuestros objetivos por eso desechamos ambas técnicas. También obviamos la técnica de "Classification rules" porque exige una tipificación previa de los usuarios, que no se dispone actualmente. Con la aplicación de la técnica de "Reglas de Asociación" buscamos las asociaciones y correlaciones entre las referencias o accesos a los archivos disponibles en el servidor web. Además, con la técnica de "Clustering" aplicada a las sesiones se intenta inferir los perfiles de usuarios, validar los patrones encontrados y descubrir nuevos.

Como han sugerido Massegkia y Cicchetti [10] las "Reglas de Asociación" pueden ser vistas como la relación entre ciertos hechos almacenados en la base de datos. Los hechos considerados pueden ser simples características o comportamientos observados en los individuos. Dos hechos pueden considerarse relacionados si ocurren para el mismo individuo. Se puede decir que una relación determinada no es relevante si ésta es observada en pocos individuos; por el contrario, si es frecuente puede considerarse un conocimiento importante para la toma de decisiones para quien intenta inferir una descripción general a partir de casos particulares.

Por otro lado la técnica de "Clustering" es la división de los datos en grupos similares de objetos, donde cada grupo es llamado cluster y contiene todos los objetos que son similares entre si y distintos a los objetos de otros grupos. La representación de los datos por medio de pocos cluster necesariamente implica la perdida de los detalles finos, pero ofrece simplificación. Esta técnica representa muchos objetos de datos por medio de pocos cluster, y por tanto, los datos son modelados a través de los cluster [1,3].

Según lo observado en la literatura de Web Usage Mining normalmente se aplican técnicas de Data Mining independientemente, en cambio en nuestro trabajo aplicamos en forma integrada dos técnicas ("Reglas de Asociación" y "Clustering"), para validar los resultados una con la otra y descubrir nuevos patrones que pueden ser mejor detectables a través de otra técnica. Cabe destacar que la primera técnica nos permitió determinar cuales eran las páginas más utilizadas tanto para ingresar como para salir del sitio, mientras la segunda fue de suma ayuda para determinar el comportamiento en general de los usuarios del portal cuando navegan a través del sitio.

4.1 Tareas de Preprocesamiento

Se realizaron varias tareas de preprocesamiento previas a la aplicación de los algoritmos de Data Mining, sobre los datos colectados en los archivos log.

Durante la tarea de **limpieza** se han verificado los sufijos del nombre del URL accedido, para identificar los elementos irrelevantes. Por ejemplo, todos los registros con extensión GIF, JPEG, JPG, y MAP pueden ser removidos por que corresponden a imágenes.

Luego de la identificación de la extensión de los archivos accedidos por los usuarios (10.983.362 registros), se ha procedido a eliminar de la base de datos aquellos que fueran indicadas como irrelevantes por el experto. Así se obtuvo un total de 828.463 registros, que fueron sometidos a su vez a otro proceso de filtrado, identificando todos los que pertenecían al Portal de Rieder. Para ello se ha utilizado la topología o estructura de páginas proveída por el Administrador del sitio web, debido a que el servidor web de Rieder también publica páginas de otras compañías clientes y registra en los archivos log los accesos a estas páginas. Realizados todos los filtros necesarios sobre los datos, se identificaron 481.915 registros validos.

A continuación se ha realizado la tarea de **identificación de los usuarios**, que permitió la identificación de todos los IP que accedieron al servidor web, por un total de 11.833 IP diferentes.

Para los archivos log que pertenecen a largos periodos de tiempo, es muy probable que el usuario haya visitado el sitio web más de una vez. El objetivo de la tarea de **identificación de las sesiones** es dividir los accesos a páginas de cada usuario en sesiones individuales. Un método simple para la identificación de la sesión es tomar todas las páginas accedidas por un usuario identificado en el archivo log y según la heurística utilizada. En nuestro caso se han agrupado los accesos en sesiones de 30 minutos como máximo entre la primera y la última página accedida. Luego se procedió a determinar la longitud del tiempo de acceso de cada página referenciada por los usuarios, mediante el calculo de la diferencia entre la hora de la siguiente referencia y la referencia actual. Para la última referencia de la sesión el tiempo se asume que es 0.
Como resultado de esta tarea fueron identificados 126.664 Sesiones de usuarios.

4.2 Identificación de transacciones

Antes de aplicar las técnicas de minería de datos, las secuencias de páginas referenciadas por los usuarios deben ser agrupadas en unidades lógicas que representen las transacciones web o sesiones de usuarios. Una sesión de usuario es el conjunto de todas las páginas referenciadas por un usuario determinado durante una simple visita al sitio. La transacción difiere de una sesión de usuario en que el tamaño de la transacción puede ser una simple página referenciada o todas las páginas referenciadas en una sesión, dependiendo del criterio utilizado para identificar las transacciones [5].
Varios autores manifiestan que los usuarios tratan a cada página a la que acceden de dos maneras: llaman referencias a *páginas auxiliares* (o *de navegación*) a aquellas utilizadas para encontrar los links a los datos, y las *páginas de contenido*, donde el usuario se detiene porque encuentra información que le resulta interesante. El campo de fecha/hora de acceso y el URL registrados en el log han sido utilizados para clasificar cada acceso como referencia de navegación o contenido para el usuario correspondiente.
Usando los conceptos de referencias a páginas de navegación y de contenido, se proponen dos maneras de definir las transacciones [1,5,12].
El primer método define una transacción como aquella que contiene todas las referencias auxiliares e incluye como última una referencia de contenido de un usuario y sesión determinados. La aplicación de las técnicas de minería de datos sobre este tipo de transacción, da como resultado los caminos más comunes de navegación para acceder a una página de contenido.
El segundo método define una transacción como aquella que contiene todas las referencias de contenido para un usuario y sesión dados, descartando las referencias auxiliares. La aplicación de las técnicas de minería de datos sobre las transacciones solamente de contenido, puede dar como resultado asociaciones entre las páginas de contenido.

5. Análisis de Patrones

De acuerdo a algunos autores [11] la cuestión fundamental en la aplicación de los algoritmos de Data Mining, es antes que nada conocer la utilidad de los algoritmos para la clase de problemas a ser considerados. En otras palabras, se debe conocer bien antes de empezar el proceso de KDD el problema particular P, con las características Cj del tipo de problema o tarea, para determinar los algoritmos específicos de Data Mining Ai que tengan mejor desempeño, para resolver el problema P.
También se requiere que el usuario interactué sobre varios pasos del KDD, especialmente cuando los resultados no son muy buenos, en términos de precisión o entendimiento (claridad) de las reglas generadas para el modelo [11].
El análisis de los Patrones es el último paso en el Proceso de Web Usage Mining. El objetivo del análisis de los patrones es filtrar las reglas o patrones que no sean interesantes dentro del conjunto encontrado en la fase de descubrimiento de patrones.
La metodología exacta de análisis está gobernada por la aplicación para la que se realiza el Web Mining. La forma más común de análisis de patrones consiste en mecanismos de consultas tal como *SQL*. Otro método es a través de cubos de datos en orden a desarrollar operaciones *OLAP* (On-Line Analytical Processing). Además, las técnicas de *Visualización*, tal como patrones gráficos o asignación de colores a diferentes valores, pueden dar una visión general de los patrones o tendencias en los datos [5,14].

5.1 Resultados obtenidos con la Técnica "Reglas de Asociación"

Analizando las reglas obtenidas con la técnica de "Reglas de Asociación", aplicada sobre los distintos archivos generados según la longitud de accesos se pueden observar ciertas similitudes entre las reglas. Los archivos fueron generados agrupando las transacciones con la misma longitud o número de accesos a páginas (referencias a páginas).
Cabe aclarar algunos términos relacionados con las Reglas de Asociación. Así el **soporte (s)** de una regla se refiere a el número de ocurrencias del conjunto de elementos en la base de datos de transacciones, mientras que la **confianza (α)** es el porcentaje de transacciones que contienen a todos los elementos que componen la regla [12].
En el contexto de Web Usage Mining, con la aplicación de esta técnica se intenta descubrir todas las asociaciones y correlaciones entre las páginas accedidas dentro de las sesiones de los usuarios, ignorando las que tienen un soporte

pobre, es decir las que no aparecen en un número suficiente de transacciones [12,14]. Para nuestro caso de estudio, cada transacción está compuesta por las páginas accedidas por un usuario dentro de una misma sesión, contemplando solo las referencias de contenido.

Archivos	Nro	Regla	Confianza
Transacciones de 3 páginas accedidas	R4	PAG1=/ PAG2=/ 422 ==> PAG3=/ 348	0.82
	R8	PAG2=usados.php 68 ==> PAG3=usados.php 49	0.72
	R13	PAG1=viewthread.php PAG3=viewthread.php 210 ==> PAG2=viewthread.php 129	0.61
	R20	**PAG2=post.php 151 ==> PAG1=viewthread.php 78**	**0.52**
	R42	PAG2=forumdisplay.php 163 ==> PAG1=viewthread.php 56	0.34
	R75	PAG1=/ 707 ==> PAG3=viewthread.php 63	0.09
Transacciones de 4 páginas accedidass	R1	PAG2=usados.php PAG4=usados.php 21 ==> PAG3=usados.php 19	0.9
	R4	PAG2=publicaraviso.php 15 ==> PAG3=publicaraviso.php 12	0.8
	R6	PAG1=u2u.php PAG2=u2u.php PAG4=u2u.php 22 ==> PAG3=u2u.php 17	0.77
	R66	**PAG2=post.php 121 ==> PAG1=viewthread.php 62**	**0.51**
	R100	PAG1=post.php 113 ==> PAG2=forumdisplay.php 18	0.16
	R192	PAG2=forumdisplay.php 127 ==> PAG1=viewthread.php 33	0.26

Tabla 1 Reglas extraídas sobre archivos que agrupan transacciones de 3 y 4 páginas accedidas

Por ejemplo se observan un par de reglas en la tabla 1, que indican que el 50% de los clientes que acceden a /post.php, accedieron antes a /viewthread.php. Esto indica que alguna información en /viewthread.php direcciona a los clientes a post.php; cabe destacar que ambas páginas pertenecen al sub sitio "La Cueva".
También se puede notar que para algunas reglas el porcentaje de confianza es elevado, mientras su representatividad es baja. Por ejemplo una regla nos indica que el 100% de los clientes que ingresan en la primera y quinta referencia en /u2u.php, también acceden a la misma página en la cuarta referencia de la transacción. Pero el número de transacciones que cumple con está regla representa el 6% de la muestra de datos, según el análisis realizado mediante consultas SQL a la base de datos.
Podemos decir que de todas las transacciones analizadas, un total de 9989, el 43% inicia la sesión en la página principal del Portal '/', el 20% en la página viewthread.php y el 7.5% en la página forumdisplay.php, el resto se distribuye uniformemente entre todas las demás páginas del portal. Esto nos indica que un buen porcentaje de transacciones se inicia a través de la página principal del Portal de Rieder, donde se encuentran los link a todos los demás sub sitios, y que otro grupo importante inicia su visita al Portal a través del sub sitio La Cueva. En la tabla 2 se presenta la relación de las transacciones con la última página accedida antes de finalizar la visita, observada en las transacciones agrupadas según el tamaño de la transacción, es decir la cantidad de páginas accedidas.

Tamaño trans.	Última Página	Porcentaje(%)
Dos páginas accedidas 4212 registros	/	32.2
	viewthread.php	16.2
	Index	10
Tres páginas accedidas 2194 registros	/	23.3
	viewthread.php	19.7
	Index	13.3
Cuatro páginas accedidas 1285 registros	viewthread.php	26.5
	/	12.5
	Index	12
	Post.php	10.5
Cinco páginas accedidas 935 registros	viewthread.php	28.3
	Index	13
	/	7.3
Seis páginas accedidas 559 registros	viewthread.php	27.5
	Index	13
	/	8.5
Siete páginas accedidas 400 registros	viewthread.php	25.7
	Post.php	16
	/	7.7
Ocho páginas accedidas 237 registros	viewthread.php	26
	Post.php	21
	/	8
Nueve páginas accedidas 114 registros	viewthread.php	35
	Post.php	12
	/	3.5
Diez páginas accedidas 53 registros	Post.php	24
	viewthread.php	22
	/	4

Tabla 2. Última página accedida en las transacciones de tamaño de 2 a 10 páginas.

Se han seleccionado las transacciones de 2 a 10 referencias o accesos, ya que representan aproximadamente el 99% de las transacciones que posibilitan la aplicación la técnica "Reglas de asociación". Observando la tabla 2 se puede notar que las páginas "/" y "viewthread.php" han sido utilizadas como última página en cerca del 50% de las transacciones de longitud 2, donde el 32 % pertenece a la página "/" principal del Portal. En las transacciones de mayor longitud el porcentaje de transacciones que utilizan la página principal disminuye regularmente, en cambio el porcentaje de transacciones que utiliza la página "viewthread.php" aumenta. Lo que significa que los usuarios que acceden a varias páginas dentro de una transacción, terminan su visita en páginas del sub sitio "La cueva".

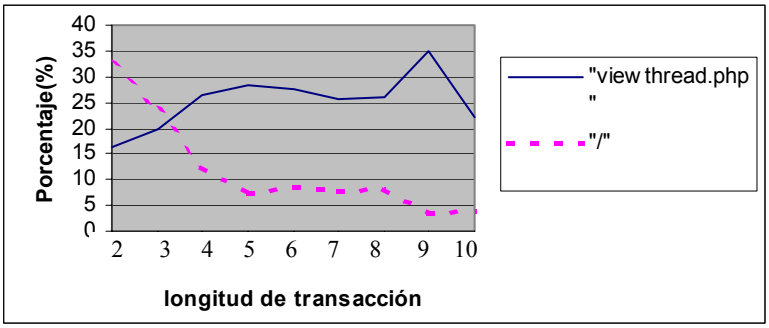


Figura 2. Última página accedida en las transacciones de tamaño de 2 a 10 páginas.

En la figura 2 se presenta la variación del porcentaje de accesos a las páginas que mayor porcentaje alcanzaron, estas son la página "/" principal del Portal y "viewthread.php".

5.2 Resultados obtenidos con la Técnica "Clustering"

Analizando los cluster obtenidos con la técnica de "Clustering", aplicada sobre los distintos archivos generados según la longitud de accesos de las sesiones, se pueden observar ciertas similitudes entre ellos.
Cabe mencionar que la estructura del sitio tiene 170 páginas diferentes, distribuidas en 6 sub sitios. Para facilitar la aplicación de la técnica de "Clustering" y el posterior análisis de los resultados, en la generación de los archivos de sesiones se reemplazaron los valores de los nombres de las páginas accedidas por los sub sitios a los que pertenecen, estos son: rieder, lacueva, elladrillo, newsletter, riederusaros, webmouse y principal para la página del Portal.
En los resultados pertenecientes a los archivos de sesiones de 2 y 3 accesos a páginas, que en conjunto representan el 70% de la muestra analizada, se puede observar que el cluster más resaltante para ambos archivos presenta un porcentaje superior al 50%, y corresponde a accesos a la página principal. Este resultado es importante porque confirma la regla encontrada en el análisis realizado con la técnica de "Reglas de Asociación" donde se descubrió que el 30% de transacciones se iniciaban en la página principal. En la Tabla 3 se presentan los cluster mencionados.

Archivo de sesiones de 2 accesos(26322)	
Cluster 1:	14327 (61%)
principal, principal	
Archivo de sesiones de 3 accesos(11075)	
Cluster 0:	5681 (51%)
principal, principal, principal	

Tabla 3. Cluster mayoritario encontrado en los archivos de sesiones de 2 y 3 accesos

Archivo de sesiones de 2 accesos(26322)	
Cluster 0:	7996 (34%)
principal, rieder	
Archivo de sesiones de 3 accesos(11075)	
Cluster 8:	2487 (22%)
principal, rieder, rieder	
Archivo de sesiones de 4 accesos(5950)	
Cluster 3:	998 (17%)
principal, principal, rieder, principal	
Archivo de sesiones de 5 accesos(3495)	
Cluster 1:	1019 (29%)
principal rieder rieder rieder rieder	

Tabla 4. Cluster encontrado en los archivos de sesiones de 2 a 5 accesos

Otra coincidencia interesante encontrada en los resultados de la aplicación de la técnica en los archivos de sesiones de 2 a 5 accesos a páginas consiste en la existencia de un cluster con un porcentaje entre 15% a 30 % de accesos tanto a la página principal como al sub sitio Rieder. Este resultado no fue observado con la técnica de "Reglas de Asociación", pero si fue descubierto como un comportamiento en un grupo importante de sesiones de usuarios. Cabe mencionar que este grupo de archivos en conjunto representa el 87% de la muestra analizada. En la Tabla 4 se presentan los cluster que contienen accesos a la página principal y al sub sitio rieder.
En los resultados de los demás archivos se ha notado una distribución cada vez más uniforme a medida que se incrementa el número de accesos por sesión.
Surgen 3 cluster importantes en el análisis de los resultados obtenidos a partir de los archivos de 6 a 10 accesos, el primero es el conjunto que incluye solo accesos a la página principal, el segundo contiene accesos al sub sitio de "La cueva" y el tercero es el conjunto de accesos al sub sitio de "Rieder". A continuación, en la tabla 4, se presentan estos resultados en detalle.
Otra coincidencia observada en los resultados de los archivos de 9 y de 10 accesos es el cluster que incluye en mayor parte accesos al sub sitio "La cueva", este comportamiento confirma los resultados obtenidos con la técnica de "Regla de Asociación", donde notamos que para los archivos de transacciones de tamaño 9 y 10 accesos, el último a página de las transacciones corresponde al sub sitio "La cueva".

Archivo de sesiones de 6 accesos (2300)	
Cluster 2:	757 (33%)
principal rieder rieder rieder rieder rieder	
Cluster 0:	656 (29%)
principal principal principal principal principal principal	
Cluster 1:	384 (17%)
lacueva lacueva lacueva lacueva lacueva lacueva	
Archivo de sesiones de 7 accesos (1606)	
Cluster 0:	511 (32%)
principal principal principal principal principal principal principal	
Cluster 9:	365 (23%)
principal rieder rieder rieder rieder rieder rieder	
Cluster 2:	296(18%)
principal lacueva lacueva lacueva lacueva lacueva lacueva	
Archivo de sesiones de 8 accesos (1153)	
Cluster 2:	284 (25%)
principal principal principal principal principal principal principal principal	
Cluster 0:	267 (23%)
principal lacueva lacueva lacueva lacueva lacueva lacueva lacueva	
Cluster 5:	225 (20%)
principal rieder rieder rieder rieder rieder rieder rieder	
Archivo de sesiones de 9 accesos (955)	
Cluster 8:	244 (26%)
principal lacueva lacueva lacueva lacueva lacueva lacueva lacueva lacueva	
Cluster 3:	184 (19%)
principal rieder rieder rieder rieder rieder rieder rieder rieder	
Cluster 5:	146 (15%)
principal principal principal principal principal principal principal principal principal	
Cluster 1:	126 (13%)
principal principal principal principal rieder rieder principal rieder rieder	
Cluster 6:	107 (11%)
principal rieder newsletter newsletter newsletter newsletter newsletter newsletter newsletter	
Archivo de sesiones de 10 accesos (772)	
Cluster 9:	128 (17%)
principal rieder rieder rieder rieder rieder rieder rieder rieder rieder	
Cluster 8:	127 (16%)
principal principal principal principal principal principal principal principal principal principal	
Cluster 2:	115 (15%)
lacueva lacueva lacueva lacueva lacueva lacueva lacueva lacueva lacueva lacueva	
Cluster 5:	114 (15%)
principal lacueva lacueva lacueva lacueva lacueva lacueva lacueva lacueva lacueva	

Tabla 5. Cluster encontrado en los archivos de sesiones de 6 a 10 accesos

6. Conclusión y trabajos futuros

En este trabajo se ha investigado y analizado la utilidad de técnicas de Data Mining aplicadas a la información Web [2,5,14], dentro del marco de Knowledge Discovery [6,11,13].
Se han presentado los pasos del proceso de Web Usage Mining y sus resultados sobre un conjunto de datos reales considerando como caso de estudio el portal de Rieder Internet, uno de los proveedores de Internet en nuestro país. Cabe destacar que, para la fase de minería de datos se adoptaron diferentes técnicas del área de Data Mining sobre el dominio especifico. En particular se utilizaron las técnicas de "Reglas de Asociación" y "Clustering".
Finalmente, con la colaboración del Experto se analizaron los patrones y reglas resultantes de la aplicación de las técnicas "Reglas de Asociación" y "Clustering" mediante el uso de los mecanismos de SQL y la técnica de "Visualización".
Entre los aportes más significativos se pueden citar:

- Propuesta de una metodología adoptando técnicas especificas de Web Usage Mining en el marco más general existente para KDD.
- Según lo observado en la literatura de Web Usage Mining normalmente se aplican técnicas de Data Mining independientemente, en cambio en nuestro trabajo aplicamos en forma integrada dos técnicas ("Reglas de Asociación" y "Clustering"), para validar los resultados una con la otra y descubrir nuevos patrones que pueden ser mejor detectables a través de otra técnica.
- El análisis de los resultados, mediante la integración de mecanismos de SQL y la técnica de "Visualización" para corroborar los patrones encontrados o descubrir nuevos.
- Además, se han presentado algunos resultados particulares obtenidos en el caso de estudio. Esto es, el descubrimiento de patrones como ser la existencia de tres grupos de usuarios con comportamientos resaltantes: el primer grupo accede solo a la página principal del Portal de Rieder, la mayoría en sesiones de corta duración (menor a 5 accesos a páginas); el segundo grupo ingresa al Portal a través de la página principal pero luego requiere páginas del sub sitio "Rieder"; y el tercer grupo accede a páginas del sub sitio "La cueva", este último grupo utiliza generalmente sesiones de larga duración (mayor a 5 accesos a páginas).

Algunas posibles áreas de interés que requieren futuras investigaciones y desarrollo son:

- Reestructuración del sitio, personalización y desarrollo de políticas de Web caching.
  En particular, utilizando información sobre las preferencias de los usuarios, que han sido obtenidas mediante técnicas de Descripción, el resultado del Web Usage Mining y técnicas de Data Mining para Predicción, se puede lograr la anticipación a la necesidad del usuario Web y proveerle información personalizada cuando visita las paginas del Portal.
- Desarrollo de Herramientas inteligentes, que puedan asistir en la interpretación de los conocimientos descubiertos, sabiendo que el resultado de los algoritmos de Data Mining generalmente no tiene un formato adecuado para el análisis de los usuarios. Claramente, este tipo de herramientas necesita conocimiento especifico sobre el dominio del problema en particular. En Web Mining, podría ser adecuado el uso de agentes inteligentes desarrollados sobre la base de los patrones de accesos descubiertos, la topología del sitio Web y ciertas heurísticas.

Bibliografía

[1]. Berkhin Pavel, "Survey of Clustering Data Mining Techniques",Accrue Software, (2002) http://citeseer.nj.nec.com/berkhin02survey.html

[2]. Cabral de Moura Borges José Luís, "A Data Mining Model to Capture User Web Navigation Patterns" . Department Of Computer Science, University College London (2000). http://www.fe.up.pt/~jlborges/publications/BorgesPhDthesis.ps.gz

[3]. Cadez Igor, Heckerman David, "Visualization of Navigation Patterns on a Web Site Using Model Based Clustering" (2000) http://citeseer.nj.nec.com/292620.html

[4]. Cooley R., Mobasher B. and Srivastava J., "Data Preparation for Mining World Wide Web Browsing Patterns" (1999). http://maya.cs.depaul.edu/~classes/ect584/papers/cms-kais.pdf

[5]. Cooley R., Mobasher B., "Web Mining: Information and Pattern Discovery on the World Wide Web" (1998) http://maya.cs.depaul.edu/~mobasher/classes/ds575/papers/Webmining.pdf

[6]. Fayyad Usama, Piatersky-Shapiro Gegory, Padhraic Smyth, "From Data Mining To Knowledge Discovery", Jet Propulsion Laboratory California Institute Of Technology. AAAI Press / The MIT Press. Menlo Park, California, USA. 1996, In ADVANCES IN KNOWLEDGE DISCOVERY AND DATA MINING, ISBN 0-262-56097-6, pag 1-34 (1996) http://www.aaai.org/Press/Books/Fayyad/Fayyad-preface.pdf

[7]. Hay Birgit, Wets Greert and Vanhoof Koen, "Clustering navigation patterns on a website using a Sequence Alignment Method" (2000) http://citeseer.nj.nec.com/451556.html

[8]. Jacobson Ivar, Booch Grady, Rumbaugh James,"El Proceso Unificado de desarrollo de Software". Rational Software Corporation. Pearson Educación, Madrid. ISBN 84-7829-036-2, pag 4-8 (2000)
[9]. Jacobson Ivar, Booch Grady, Rumbaugh James, "El Lenguaje Unificado de Modelado". Rational Software Corporation. Addison Wesley Iberoamericana, Madrid ISBN 84-7829-028-1. (1999)
[10]. Massegkia Florent, Ponclet Pascal, Cicchetti Rosine, "An efficient algorithm for Web usage mining", Universite de Versilles, Francia. (2000)
http://citeseer.nj.nec.com/399609.html
[11]. Meneses Claudio J. , Grinstein Georges G.,"Categorization And Evaluation Of Data Mining Techniques", Departamento de Ing. de Sistemas y Computación, Universidad Católica Del Norte, Antofagasta-Chile . Proc. of Intl. Conf. on Data Mining, Sept. 1998, In DATA MINING (Ebecken, N.F.F., editor), ISBN 1853126772 pag. 53-80 (1998)
[12]. Mobasher B. and Srivastava J., "Web Mining: Pattern Discovey from World Wide Web Transactions" (1996)
http://citeseer.nj.nec.com/mobasher96web.html
[13]. Rezende S.O., Oliveira R.B.T, "Visualization for Knowledge Discovery in Database", Departament of Computer Science and Statistics, Institute of Mathematical and Computer Sciences, University of Sao Paulo.1996, In DATA MINING (Ebecken, N.F.F., editor), ISBN 1853126772 pag. 81-95 (1996)
[14]. Srivastava Jaideep, Cooley Robert, Deshpande Mukund, Tan Pang-Ning, "Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data", Department of Computer Science and Engineering. University of Minnesota(2000)
http://www.cs.umn.edu/research/websift/papers/sigkdd00.ps

A Semantics Definition Metamodel

Ma. Laura Caliusco y César Maidana
GIDSATD - UTN - FRSF
Lavaise 610 – Santa Fe
Santa Fe - Argentina
+54 0342 4608585 – Int 222
{mcaliusc, cmaidana}@frsf.utn.edu.ar

y

Ma. Rosa Galli y Omar Chiotti
INGAR-CONICET
Avellaneda 3657
Santa Fe - Argentina
+54 0342 4534451
{mrgalli,chiotti}@ceride.gov.ar

Abstract
New information technologies provide new opportunities for allowing collaborative business-to-business (B2B) relationships. An effective B2B relationship requires a right modeling of collaborative processes and each message within these processes.
The XML (eXtensible Markup Language) is becoming widely used for representing both the processes and the business documents. However, the use of XML is insufficient for implementing an effective B2B relationship because of semantics heterogeneity that takes place in a collaborative process.
In order to represent semantics ontology specification languages from Artificial Inteligence (AI) area have arisen. However, the main disadvantage of these languages is they are mostly based on logic formalisms to support machine reasoning. This makes the language syntax unfamiliar for business analysts who define the collaborative process and model the business documents to be exchanged.
To fill the gap between people involved in the business documents definition and ontology specification languages, there are some proposals for the use of the Unified Modeling Language (UML) in ontology development. But, UML does not satisfy all requirement for ontology modeling.
In this paper we present a metamodel for ontology definition. The objective of this metamodel is to overcome the gap between B2B area and AI techniques to model semantics associated to XML-based B2B documents.

Keywords

Ontology, Metamodel, XML, business-to-business, aritificial intelligence, software engineering.

1. INTRODUCTION

New information technologies, such as Internet, web-based applications and distributed systems provide new opportunities for allowing collaborative business-to-business (B2B) relationships. In these relationships companies can operate as a single entity and make joint decisions focusing on adding value for their customers. An effective B2B relationship requires a right modeling of collaborative processes and each message within these processes. Each message contains business information that may be defined by a vocabulary that is shared by the parties engaged in the B2B relationship. This business information is contained by a business document, like Purchase Order, Catalog and so on.

The XML (eXtensible Markup Language) is becoming widely used for representing both the process and the business documents [6]. XML was created to facilitate data interchange among different applications, data sources, and operating systems. Due to the data within an XML document are tagged, the document carries with the information necessary to recognize, extract, and manipulate those data. Many XML-based specifications for B2B area have been defined, such as RossettaNet[1], ebXML[2] and OAGIS[3]. However, the use of XML is insufficient for implementing an effective B2B relationship because of semantics heterogeneity [4].

According to Uschold (2003), semantics means something that carries meaning and can be implicit, explicit and informal, explicit and formal for human processing and explicit and formal for machine processing. He defines a semantic continuum applying these concepts to Semantic Web. In Figure 1 we represent this idea for XML-based B2B specifications. In Figure 1 (a), it can see that XML specifications contain implicit semantics due to the meaning of each tag exists only in the minds of the humans who have defined them. For example, the tag "*ItemQuantity*" has only meaning for the persons who have defined it and its meaning is implicitly encoded in the application that use it. There are specifications that contain explicit semantics informally defined. For example, OAGIS specifications contain explicit semantics defined between *annotation* elements in natural language, as it can see in Figure 1 (b). This helps the implementation of B2B XML-specifications but it is not enough for determining in unambiguous way the meaning of the data at run time [4].

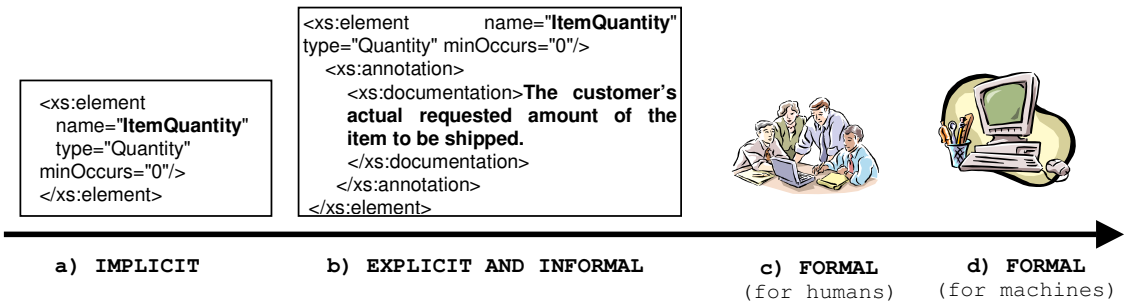


Figure 1. Semantic Continuum.

To process the semantics at run time, it has to be expressed in a machine processsable language. A lot of work has been done in the Semantic Web area, where computers will be able to use the data on the web not just for display purposes, but for automation, integration and reuse of data across various applications [2]. Klein (2003) has defined a procedure that can be used to turn XML documents into knowledge structures specified in an ontology specification language for Internet. From B2B perspective, the main disadvantage of ontology specification languages is they are mostly based on logic formalisms to support machine reasoning. This makes the language syntax unfamiliar for business analysts who define the collaborative processes and model the business documents to be exchanged. Furthermore, the collaborative processes and the business documents have to be implemented by software engineers who have to interpret the information model.

To fill the gap between people involved in the business documents definition and ontology specification languages, there are proposals for UML use in ontology development [8]. But, UML itself does not satisfy needs for representation of ontology concepts that are borrowed from Descriptive Logic and that are included in ontology specification languages [9].

The objective of this paper is to present a metamodel for modeling explicit and formal semantics, for human processing, associated to XML-based business documents. Firstly, we analyze the UML role in ontology modeling.

[1] www.rosettanet.com
[2] www.ebxml.org
[3] www.openapplications.org

Then, we present a metamodel for ontology modeling using "Unified Modeling Language: Infrastucture" specification, version 2 [17] and analyze the relationship between the XML specifications and ontologies in order to add formal and explicit semantics, for human processing, to business documents. Finally, we present future directions and conclusions.

2. UML FOR ONTOLOGY MODELING

The Unified Modeling Language (UML) is a standard defined by the Object Management Group [14]. UML defines an abstract language for describing the structure and behavior of software systems. A standard graphical notation is also defined for creating views of the model elements in this language.

UML has been used to model ontology due to UML class diagram can be used to express concepts in term of classes and relationships among them. Cranefield (2001) proposed an ontology representation formalism based on a subset of the UML together with its associated Object Constraint Language (OCL) for agent software communication.

One advantage of using UML for ontology modeling is that it is easy to understand for business experts. Another advantage is that there are many tools for creating and editing UML models that can be used for ontology modeling. However, UML and OCL have some limitations to represent an ontology. In the next section we briefly discuss these limitations.

2.1 UML and OCL Limitations

The main obstacle for using UML as an ontology modeling language is that the notion of Property in UML metamodel is not the same as the notion of Property in ontology [10]. *Property* in ontology corresponds to the notion of *association* in UML.

In addition, UML has not the appropriate support for describing restrictions. To represent constraints and restrictions into an UML diagram OCL could be used. However, it is difficult to translate axioms to other languages due to OCL has not explicit and unambiguous semantics. For example, a constraint may be encoded by several expressions [8].

Some approaches propose extending UML metamodel to tackle these problems adding the notion of *Property* and *Restriction* as UML *Classifier* [1]. The inclusion of this extension into UML specification causes an important impact on existing tools because it implies to change the data model of these tools.

Furthermore, UML does not provide a formal way to model relations between concepts from semantic point of view. For example, if we want to express that two concepts are synomyns we have to describe this relation by using stereotypes and add axioms with comments in natural language which is ambiguous.

So, instead of extending UML class diagrams to represent information semantics we propose to define a Metamodel based on MOF. In the next section, we analyze the propused metamodel and following an example we use it to model a semantics associated to an XML-based business document.

3. A METAMODEL FOR ONTOLOGY MODELING

The use of ontologies to solve the semantic problem in business integration is not new. However, quite often ontologies are used as simple or structured vocabularies and in this role they do not provide any substantial benefit comparing to existing techniques (Omelayenko, 2002). The most commonly used definition, offered by Grüber (1993) states that: ''an ontology is an explicit specification of a conceptualization''. In this context, it should be clear that "explicit" object is a concrete, symbol-level object. But "conceptualization" is not clear and sometimes "conceptualization" is defined as an abstract model of some phenomenon in the world by having identified the relevant concepts of that phenomenon.

An ontology can be classified as either a lightweight or heavyweight ontology. On the one hand, lightweight ontologies include concepts, concept taxonomies, relationships between concepts and properties that describe concepts. On the other hand, heavyweight ontologies add axioms and constraints to lightweight ontologies [7].

In B2B area, it is common to view lightweight ontologies that describe simple taxonomies of products, catalogs and purchase orders. Our purpose is to define a metamodel that assists to business analysts in the modeling of more enrich ontologies. In order to define the metamodel we have used the Core Package of the "Unified Modeling Language: Infrastucture" specification, version 2 [17].

First, we have to make clear what we have in mind when we refer to ontology. An ontology can be defined as a set of concepts. Concepts imply a set of *terms* and *relations* between them. Furthermore, it is suitable to add *properties* and *axioms* to enrich the ontology. The set of properties define the characteristics of *terms*. *Axioms* are properties of the relation between ontology terms. For example, *PurchaseOrder* is a sub-class of *Order*, and this relation is not a symmetric one (axiom). Following we formalize this definition.

Definition 1. *An ontology O_i is a 4-tuple $<T_i, P_i, R_i, A_i>$ where:*
i identifies the domain or source that an ontology is associated with.
T_i is a set of terms t_j of O_i
P_i is a set of properties of terms $t_j \in T_i$
R_i is a set of relations between t_j and $t_x \in T_i$
A_i is a set of axioms that characterizes each relation of R_i

The typical role of a metamodel is to define the semantics for how a model elements in a model gets intantiated. The main design principles of the propused metamodel are: easy of use in rapid development of ontologies from business documents (standard or ad-hoc) and modularity.

In order to fill the modulariy design principle, the metamodel constructs were grouped into packages according to the elements needed to define an ontology. These packages are:

1. *Kernel Package*: contains classes and associations that form the kernel of the metamodel, wich are used by all other packages. This package imports and specializes elements from InfrastructureLibrary::Core [17].
2. *DataType Package*: contains classes and associations that can be used to create data types and data values for use in defining an ontology.
3. *Ontology Package*: contains classes and associations that can be used to define an ontology.
4. *Terms and Properties Package*: contains classes and associations that can be used to model terms and their properties.
5. *Relations Package*: contains classes and associations that can be used to model relations between terms belonging to an ontology.
6. *Axioms Package*: contains classes and associations that can be used to describe axioms about relations.

Following we define the last four packages and present some issues to model the semantics associated to a XML-based business document. This business document, showed in Figure 2, represents the information that a customer sends to the supplier in order to agree on a supply plan. The XML element *documentation* allows us to instantiate an Annotation class from the metamodel and to associate it to the corresponding element.

```
1) <xs:complexType name="PlanningSchedule">
2)     <xsd:element name = "Date" type = "xsd:date"/>
3)     <xsd:simpleType name = "Description">
4)       <xsd:restriction base = "xsd:string">
5)             <xsd:maxLength value = "40"/>
6)       </xsd:restriction>
7)    </xsd:simpleType>
8)    <xsd:element name = "Item" type = "Items"/>
9)    <xs:element name="ItemQuantity" type="Quantity"  minOccurs="0"/>
10)      <xs:annotation>
11)         <xs:documentation> The customer's   actual requested amount of the item to be shipped.
13)         </xs:documentation>
14)      </xs:annotation>
15)   </xs:element>
16)   <xsd:complexType name = "LocalAddress">
17)       <xsd:complexContent>
18)        <xsd:extension base = "Address">
19)          <xsd:element name= "zip" type = "xsd:string"/>
20)        </xsd:extension>
21)       </xsd:complexContent>
22)    </xsd:complexType>
23) </xsd:complexType>
```

Figure 2. XML-based business document.

3.1 Ontology Package

In Figure 3 are represented the classes and associations of the Ontology Package needed to model an ontology.

The *Element* is an abstract metaclass with no superclass. An element is a constituent of a model. The *NamedElement* is an abstract metaclass that represents elements that may have a name. The name is used for identification of the named element within the namespace in which it is defined. These classes are defined in the UML infrastructure library [17].

The main component of this package is *Ontology* class that includes definition of concepts used to describe and represent a domain. This class is associated with the class *OntologyElements* which is an abstract metaclass that gruops the objects of an ontology metamodel. If an ontology is removed, so are the elements owned by it. The association *imports* represents that an ontology could contains definitions whose meaning are defined in other ontologies. The association *prior_Version* identifies the referred ontology as a prior version of one ontology.

Each ontology element could be describe by a comment, represented by the *Comment* class. The *Body* atribute specifies a string that is the comment. This class intend to model, for example, the *xsd:annotations* elements from XML-based documents.

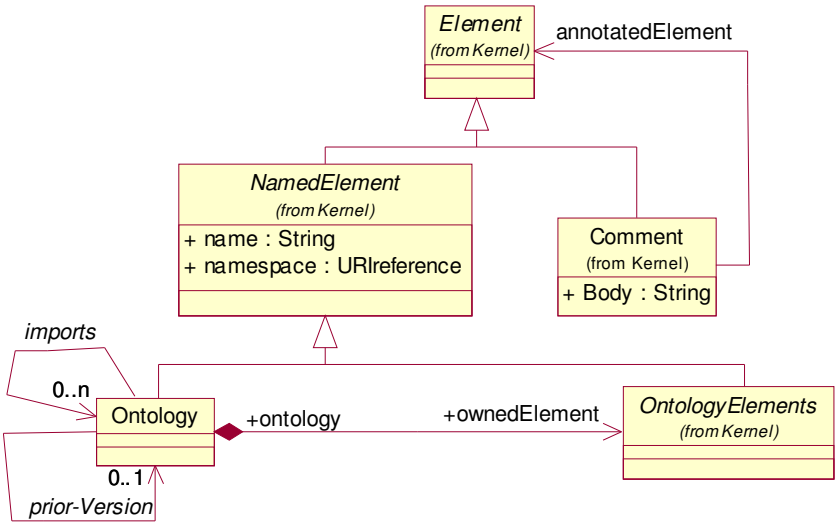


Figure 3. Ontology package.

3.2 Properties and Terms Package

Terms represent the set of concepts that a business analyst wants to represent in an ontology. A term can be simple or complex. A simple term has associated a value and a complex term is composed by other terms. We can state that $T_i = T_i^S \cup T_i^C$, where T_i^S is the set of simple terms and T_i^C is the set of complex terms.

Properties describe the features of a term. For example, allowed values, the number and other features of the values that a simple or complex term could take.

The metamodel that represents the relation between *Properties* and *Terms* is presented in Figure 4. In the proposed metamodel, the class *Properties* defines the features of a term so this class has to be associated at least one instance of *Terms*. It is indicated with the label 0..1 in the association end.

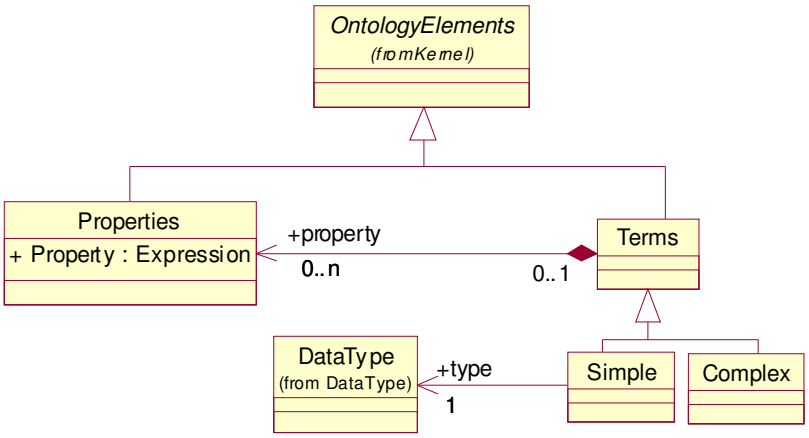


Figure 4. Properties and Terms Package.

Following we present some issues to model the terms belonging to a XML-based business document:

1. The *xsd:complexType* element has to be model by the *Complex* class. For example, from the document defined in Figure 2, line 1, we can model "**PlanningSchedule**" and "**LocalAddress**" terms as complex one.
2. The *xsd:simpleType* element has to be model by the *Simple* class. For example, from the document presented in Figure 2, line 3, we can model "**Description**" element as a simple term.
3. Then the elements defined by *xsd:element* tag could be simple or complex depending on its type definition. That is, if they are defined as a base xsd type, they are simple terms. For example, in line 2 the "**Date**" element is defined as "xsd:date" and in line 19 the "**zip**" element is defined as "xsd:string". Both elements have to be modeled as simple terms. Then, in line 8 and 9, the "**Item**" and "**ItemQuantiy**" elements are defined as *Items* and *Quantity* respectively. In order to determine if they are simple or complex terms we have to analyze if *Items* and *Quantity* have been defined as simple or complex terms. These difinitions are not in this document. So, we suppose that *Items* was defined as complex term and *Quantity* was defined as simple one.
4. Furthermore, in this document (line 3) the "**Description**" element is restricted by using xsd:restriction definition. This characteristic has to be modeled as a property of the "**Description**" term. That is, the xsd:restriction element has to be modeled by *Properties* class.

Figure 5 presents the model of the terms belonging to the XML-based document represented in Figure 2. This model was obtained by applying the issues defined aboved. The notation used to graphically represent this model is based on the UML notation by using UML stereotypes.

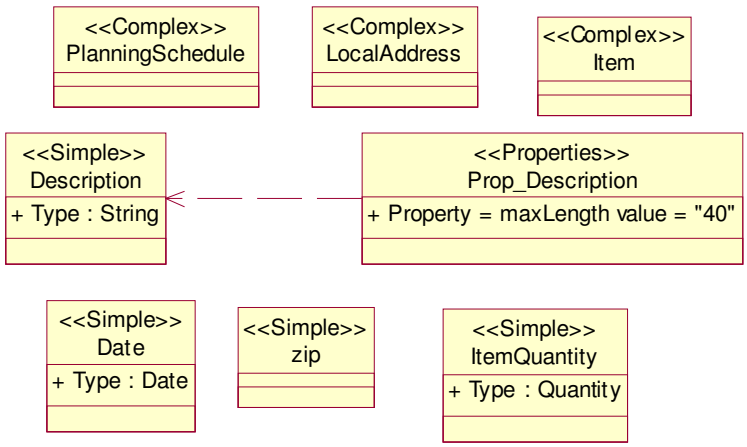


Figure 5. Model of business document terms.

3.3 Relations Package

Relations R_i represent how terms belonging to O_i are related to. Relations can be divided into hierarchical relations (R_i^H), conceptual relations (R_i^C) and particular relations (R_i^P). That is, $R_i = R_i^H \cup R_i^C \cup R_i^P$. We do not put any restrictions on the set R_i^P because it intends to model the relations defined by the ontology modeler. Whereas we require that $R_i^H = \{\text{is-a, part-of, inst-of}\}$. Furthermore, we propose that $R_i^C = \{\text{synonym, antonym}\}$, but it can be extended to add other conceptual characteristics.

The relations metamodel is presented in Figure 6. *Terms* and *Relations* classes are associated via the *RelationEnd* class. An instance of *Relations* class has to be associated at least with two instances of *RelationEnd* class, it is indicated with the label 2..n. *RelationEnd* class is associated with one *Terms* class and contains the information about cardinality and the role of terms. Furthermore, this class has the *Navigable* attribute to represent the direction of the relation.

One important ontologies requirement is its ability to structure the relations into hierarchies. That is, to define sub-relations of a relation. Furthermore, it is suitable to define equivalent relations and inverse relations. These are modeled by the relations *subrelationof*, *inversof* and *equivalentto*.

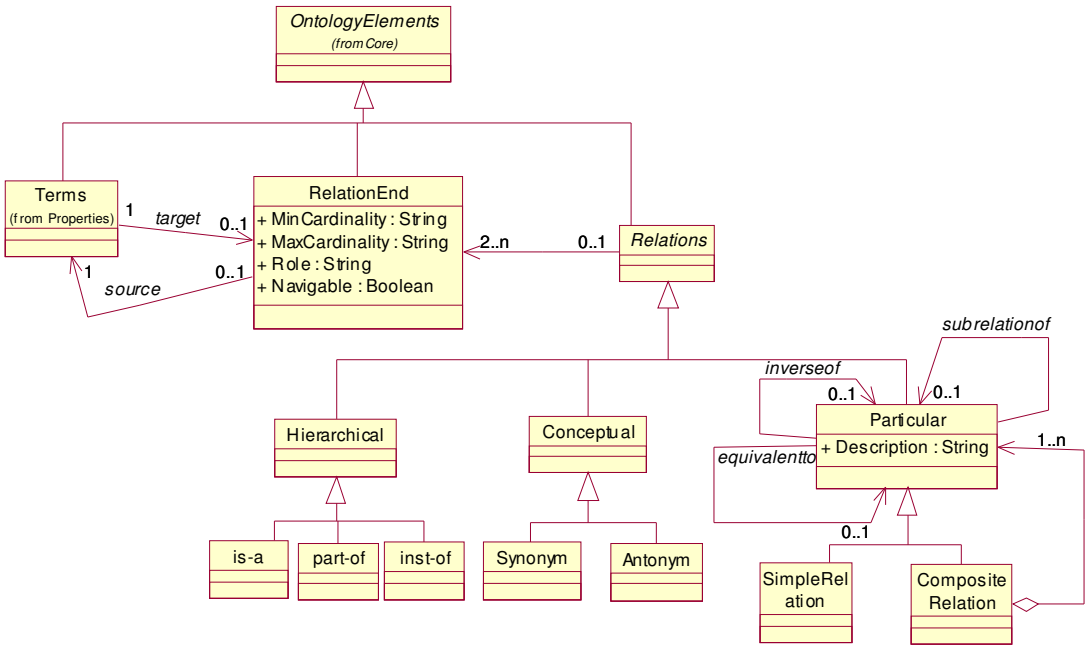


Figure 6. Relations Package.

The R_i^H set can be derived from an XML document [3]. Figure 7 represents the model of terms and relationships between them after applying the following rules to the XML business document presented in Figure 2.

1. The *xsd:extension* element represent the *is-a* relationship between terms. For example, in Figure 2 line 18, the <xsd:extension base = "**Address**"> definition state that the element previously defined (**LocalAddress**) *is-a* **Address**.
2. The combination of both *complexType* and *element* primitives represents the *part-of* relationship between terms. For example, all elements defined into the *complexType* primitive that define the **PlanningSchedule** term are related with it by the *part-of* relation. That is, **Date**, **Description**, **Items**, **ItemQuantity** and **LocalAddress** are *part-of* **PlannningSchedule**. Furthermore, **zip** is *part-of* **LocalAddress**.
3. The *element* primitive represents the *Inst-of* relation between terms.

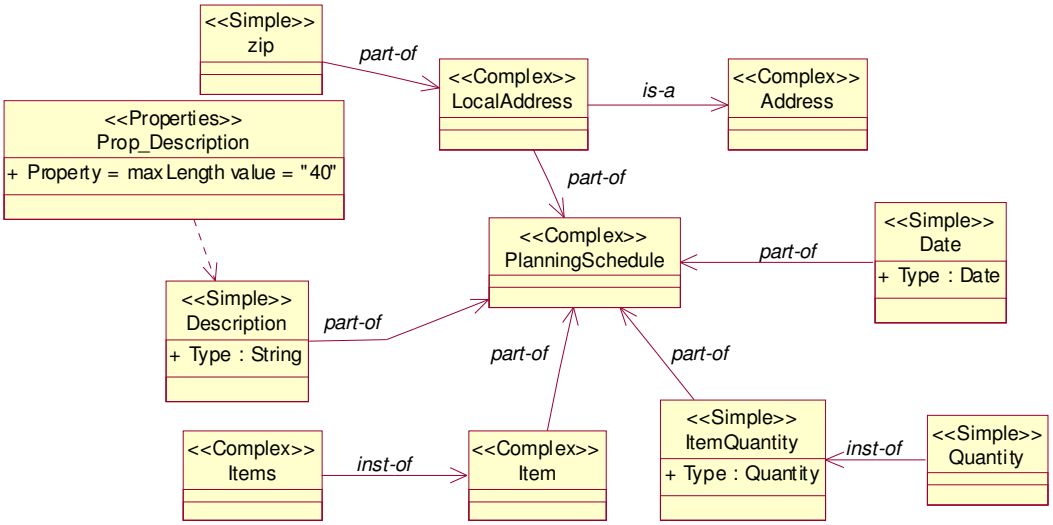


Figure 7. Model of terms and relationships between them.

3.4 Axioms Package

Axioms are properties of relations and they help to constraint the concepts interpretation. Furthermore, they provide guidelines for automated reasoning. In the knowledge engineering area, axioms have been represented using logic languages. In the UML class diagram, axioms could be expressed by OCL. For example, OCL constraints have to be used to declare a transitive property of a relation between terms. However, describing such constrains may involve writing moderately complex OCL expressions that are not immediately understandable to a human reader. In addition, there may be several different expressions encoding the same constraint. An interesting issue is to represent axioms as objects [16].

Axioms can be divided into several subsets. In this work we define $A_i = A_i^{RA} \cup A_i^P$, where A_i^{RA} represents the set of axioms for relational algebra and A_i^P represents the set of particular axioms, which are defined by the users. We do not put any restrictions on the set A_i^P, whereas we required that A_i^{RA} ={symmetric, reflexive, transitive, functional}.

Figure 8 represents the metamodel for modeling axioms and their association with the *Relations* class. The *AParticular* class is related to *Period* class for modeling temporal axioms.

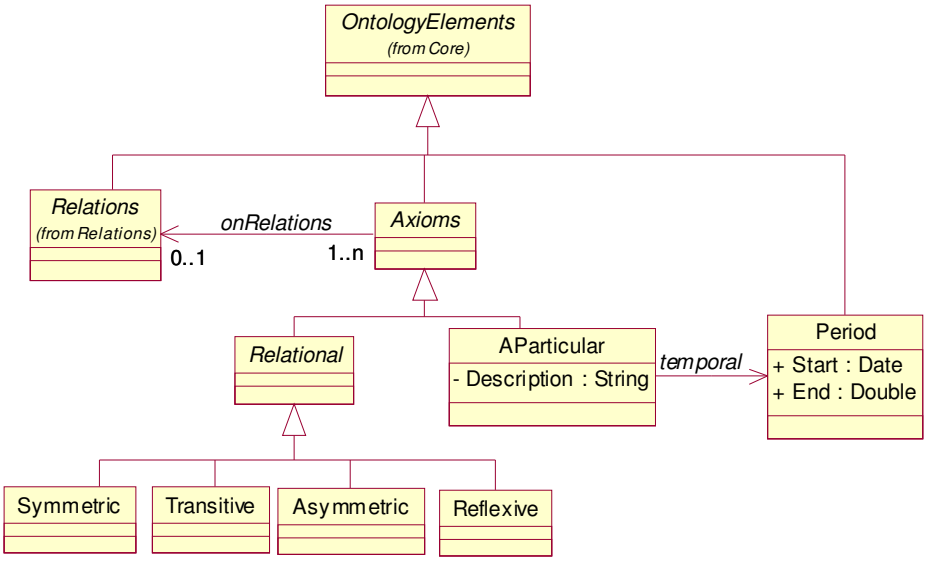


Figure 8. Axioms Package.

4. CONCLUSIONS AND FUTURE DIRECTIONS

To effectively carry out a collaborative B2B relationship, enterprises need to unambigously understand the business information that they interchange. To do that, they have to implement an ontology in order to represent the semantics associated to this information.

The semantics associated to B2B specifications has to evolve by moving along the semantic continuum from implicit semantics to formal semantics in order to support an effective information processing. The metamodel defined in this paper allow to people involved in a B2B relationship modeling the semantic associated to XML-based business documents. The concepts defined in this metamodel for modeling ontology can be used as stereotypes in UML. Furthermore, we have defined a set of rules to model the semantics implicit in XML-based business documents.

The future directions will be focus on the mapping between this metamodel and the ontology specification languages. There are traditional ontology specification languages and other languages created in the context of Internet that exploit the characteristics of the Web. Such languages are usually called *web-based ontology languages* or *ontology markup languages*. These languages are still in a development phase: they are continuously evolving [7].

The more recently defined language is OWL Web Ontology Language [13] which is a W3C (World Wide Web Consortium) proposed recommendation. So, we are interesting in the translation problem between the metamodel propused in this paper for ontology modeling and this language.

REFERENCES

[1] Baclawski, K., Kokar, M., Kogut, P., Hart, L., Smith, J., Holmes,W., Letkowski, J., Aronson, M.: Extending UML to Support Ontology Engineering for the SemanticWeb. In *Fourth International Conference on UML*, Toronto (2001)

[2] Berners-Lee, T.; Hendler, J. and Lassila, O. The Semantic Web. Scientific American **284** (May 2001), 35–43. Available on-line http://www.scientificamerican.com

[3] Bouquet, P, Dona, A, Serafini, L and Zanobini, S. Contextualized local ontologies specification via CTXML. AAAI-02 Workshop on Meaning Negotiation (MeaN-02) July 28, 2002, Edmonton, Alberta, Canada. Available on-line http://sra.itc.it/people/serafini/distribution/aaai-ws-ctxml.pdf

[4] Caliusco, Ma. Laura, Galli, Ma. Rosa and Chiotti, Omar. Ontology and XML-based specifications for collaborative B2B relationships. In *Proceeding of III Jornadas Iberoamericanas de Ingeniería de Software e Ingeniería de Conocimiento (JIISIC)*. Valdivia (Chile). November 2003.

[5] Caliusco, Ma. Laura, Galli, Ma. Rosa and Chiotti, Omar. Ontology Role in Collaborative Business-to-Business Relationships. In *Proceeding of the XXIX Conferencia Latinoamericana de Informática (CLEI 2003)*. La Paz, Bolivia. September, 2003.

[6] Carlson, D. Modeling XML Applications with UML – Practical e-Business Applications. Addison Wesley, 2001.

[7] Corcho, O.; Fernández-López, M.; Gómez-Pérez, A. *Methodologies, tools and Languages for building ontologies*. Where is the meeting point? Data and Knowledge Engineering. 46 (2003) 41–64.

[8] Cranefield, S., Haustein, S. and Purvis, M.. UML as an ontology modelling language. In *Proceedings of the Workshop on Ontologies in Agent Systems, 5th International Conference on Autonomous Agents, Montreal, Canada, 2001.*

[9] Duric, D.; Gasevic, D; Devedzic, V. *A MDA-based Approach to the Ontology Definition Metamodel*. Proceedings of a 4TH Workshop On Computational Intelligence And Information Technologies. October 13, 2003, Faculty of Electronics, Niš, Serbia.

[10] Falkovych, K., Sabou, M., Stuckenschmidt, H.. *UML for the Semantic Web: Transformational Approaches* in B. Omelayenko and M. Klein: Knowledge Transformation for the Semantic Web, IOS Press, 2003.

[11] Grüber, T.R. (1993) A translation approach to portable ontology specification, *Knowledge Acquisition* 5 199–220.

[12] Klein, M. *Interpreting XML via an RDF schema*. chapter in *Knowledge Annotation for the Semantic Web* IOS Press, Amsterdam, 2003.

[13] McGuinness, D and van Harmelen, F. OWL Ontology Web Language – Overview. W3C Recomended Propesed. December 2003. Available on-line http://www.w3.org/TR/owl-features/

[14] Object Management Group. Unified Modeling Language. http://www.uml.org

[15] Omelayenko B. Ontology-Mediated Business Integration. In *Proceedings of the 13-th EKAW 2002 Conference*, Siguenza, Spain, October 1-4, LNAI 2473, 2002, pp. 264-269 © Springer-Verlag

[16] Staab, S; Mädche, A.. Axioms are objects, too - Ontology Engineering Beyond the Modeling of Concepts and Relations. In: V.R. Benjamins, A. Gomez-Perez, N. Guarino (eds.). *Proceedings of the ECAI 2000 Workshop on Ontologies and Problem-Solving Methods*. Berlin, August 21-22, 2000.

[17] UML 2.0 Infrastructure – Final Adopted Specification. September, 2003.

[18] Uschold, M. *Where is the semantics in the Semantic Web?* **AI Magazine.** Volume 24 , Issue 3 (September 2003) Pages: 25 – 36. ISSN:0738-4602. Disponible on-line lsdis.cs.uga.edu/SemWebCourse_files/WhereAreSemantics-AI-Mag-FinalSubmittedVersion2.pdf (2003).

Algoritmos para el problema de las n-reinas

Alfredo Candia Véjar
Universidad de Talca, Dpto. de Ingeniería de Sistemas,
Curicó, Chile
acandia@utalca.cl

and

César Astudillo Hernández
Universidad de Talca, Dpto. de Ingeniería de Sistemas,
Curicó, Chile
castudillo@utalca.cl

Abstract

The n-queens problem is to find the different ways to assign n nonattacking queens in a $n \times n$ chessboard. This work analizes the application of an algorithm based on Local Search for the resolution of the n-queens problem.

Empirical results show that, large size instances of the problem are well solved by the Local Search algorithm in comparison to more sofisticated algorithms like Genetic Algorithms.

Keywords: n-queens problem, Local search.

Resumen

El problema computacional de las n-reinas consiste en encontrar las diferentes formas de asignar n reinas a un tablero $n \times n$, de manera que éstas no se ataquen. Este trabajo analiza la aplicación de algoritmos basados en Búsqueda Local para la resolución del problema de las n-reinas.

La experimentación computacional efectuada con instancias de gran tamaño muestra que se obtienen interesantes resultados con el algoritmo de Búsqueda Local en comparación con algoritmos más sofisticados tal como Algoritmos Genéticos.

Palabras claves: Problema de las n-reinas, Búsqueda Local.

1. INTRODUCCION

El problema de las n-reinas (**N-Q**) consiste en encontrar una asignación de n reinas a un tablero $n \times n$ de modo que éstas no se ataquen. El problema (para $n = 8$) fue propuesto en el año 1848 en un trabajo anónimo [2], y que posteriormente fue atribuído a Max Bezzel. La publicación detallada más antigua que se conoce fue realizada por Nauck en 1850 [15]. Ese mismo año, según unas cartas publicadas en 1929, Gauss conjeturó que existían 72 soluciones para $n = 8$; en el año 1874, Glaisher [7] probó que existían en realidad 92 soluciones. La investigación sobre este tema no ha cesado hasta hoy y luego existe una amplia variedad de algoritmos sugeridos para su resolución.

N-Q tiene dos versiones. La más simple consiste en encontrar exactamente una solución al problema para un valor dado de n. La versión más difícil consiste en encontrar todas las soluciones para un valor dado de n. Notamos que N-Q tiene una solución para $n = 1$, no tiene para $n = 2$ y $n = 3$ y tiene dos soluciones para $n = 4$ (ver fig. 1). Para mayores valores de n, N-Q tiene una función estrictamente creciente de soluciones y más aún su crecimiento es exponencial siguiendo un análisis empírico. Algunos matemáticos han estudiado el problema de encontrar todas las soluciones y hasta ahora no han encontrado una fórmula cerrada para tal problema de enumeración. La Tabla 1 ilustra el número total de soluciones $Q(n)$ para $4 \leq n \leq 20$ [16].

Numerosos algoritmos se han diseñado para N-Q tales como backtracking, algoritmos genéticos, búsqueda local con resolución de conflictos, programación entera, dividir para conquistar y redes neuronales, entre las más conocidas. N-Q pertenece a la clase de problemas computacionales NP-completos [4], pero se resuelve

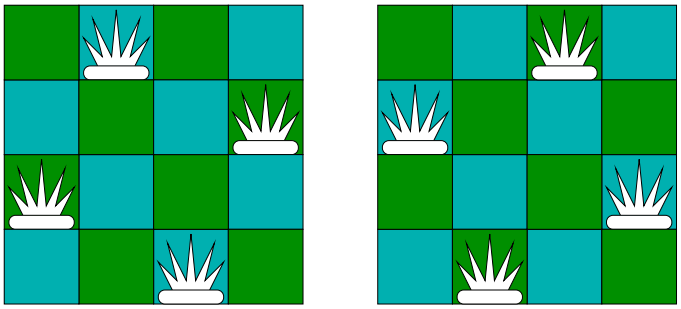


Figura 1: Soluciones para n=4

n	$Q(n)$
4	2
5	10
6	4
7	40
8	92
9	352
10	724
11	2.680
12	14.200
13	73.712
14	365.596
15	2.279.184
16	14.772.512
17	95.815.104
18	666.090.624
19	4.968.057.848
20	39.029.188.884

Tabla 1: Número total de soluciones $Q(n)$ para $4 \leq n \leq 20$

fácilmente, en tiempo polinomial observado, cuando se requiere solamente una solución [17]. Existen soluciones analíticas para N-Q donde se da una fórmula explícita para la localización de las reinas o bien se encadenan soluciones obtenidas para valores menores de n [3][1]. El problema con estas soluciones es que generan un número muy pequeño de soluciones. Backtracking, en general es ineficiente y para N-Q no es fácil encontrar soluciones para $n > 100$ en tiempos razonables, ver Stone y Stone [18], sin embargo, Kalé [13] diseñó un algoritmo especializado de backtracking que consigue resolver el problema hasta tamaños del orden de 1.000. Se han realizado numerosas aplicaciones de algoritmos genéticos a N-Q [5], [4], [9], [10], pero se han visto ineficientes aún cuando se hayan probado diversas representaciones del problema y diversos operadores. Algoritmos para Programación Entera son típicamente exponenciales y consecuentemente consiguen resolver problemas de tamaño muy pequeño [6]. Un algoritmo de redes neuronales utilizando un modelo modificado de Hopfield [11] parece funcionar mejor que otros modelos de redes neuronales presentados [14][12] , pero se reportan resultados solamente para valores muy pequeños de n. Búsqueda local con minimización de conflictos [17] parece ser el mejor algoritmo encontrado hasta ahora. Este algoritmo de Sosic y Gu tiene tiempo de corrida (observado) lineal y luego es extremadamente rápido. En su artículo, ellos muestran que su algoritmo es capaz de obtener una solución para cualquier valor de n, $n < 1000$ en menos de 0.1 segundo, y 55 segundos para n = 3,000,000, en una máquina rápida del año 94 (IBM RS 6000/530). Dada la naturaleza probabilística del algoritmo, no se tiene una garantía del tiempo del peor caso del algoritmo pero exhibe en la práctica un excelente desempeño y un comportamiento muy robusto. Hynek [10] diseñó una heurística basada en una fase de preprocesamiento y posteriormente aplica un operador de mutación basado en búsqueda local. Los resultados computacionales muestran que el algoritmo es efectivo y logra soluciones hasta $n = 1000$ en tiempos razonables.

N-Q es conocido usualmente como un problema relacionado a un juego y también como un problema apropiado para probar algoritmos nuevos. Sin embargo, N-Q tiene también algunas aplicaciones; de hecho, se le considera como un modelo de máxima cobertura. Una solución a N-Q garantiza que cada objeto se puede

accesar de cualquiera de sus ocho direcciones vecinas (dos verticales, dos horizontales, y cuatro diagonales). Algunas aplicaciones posibles son: control de tráfico aéreo, sistemas de comunicación modernos, programación de tareas computacionales, procesamiento paralelo óptico, compresión de datos y balance de carga en un computador multiprocesador, ver [17] para mayores detalles.

La sección 2 discute representaciones para el problema, la definición de posibles vecindades a usar en algoritmos basados en búsqueda local y la definición de la función de costo a utilizar. La sección 3 describe un algoritmo de búsqueda local simple para N-Q y se discuten resultados computacionales para diversos valores de n. Adicionalmente, se discute la problemática de obtener un conjunto de soluciones diferentes mediante la aplicación repetida de este algoritmo y se compara con el algoritmo aleatorizado de Gu [17]. Finalmente, comentamos algunas mejorías al algoritmo propuesto y también presentamos una extensión del problema.

2. REPRESENTACION, VECINDADES Y COLISIONES

Una representación natural para N-Q es aquella de utilizar matrices para representar el tablero $n \times n$. En este caso, un elemento de la matriz es un número $a_{ij}, 1 \leq i, j \leq n$, donde $a_{ij} \in \{0, 1\}$, $a_{ij} = 1$, si una reina se localiza en la posición i, j, y $a_{ij} = 0$ en caso contrario. Evidentemente que si localizamos al azar n reinas representadas por el valor 1 en la matriz A, entonces pueden aparecer colisiones horizontales, verticales y diagonales. Con el objetivo de trabajar solamente con colisiones diagonales, y dado que queremos minimizar el número de colisiones (una solución debe tener 0 colisiones) es más usada la representación denominada *permutación*. En una permutación, usamos un vector de longitud n, donde cada posición j $(1 \leq j \leq n)$, contiene un número a_j indicando que existe una reina en la posición (j, a_j). Por ejemplo, para $n = 4$, el vector $(2, 4, 1, 3)$ indica que existe una reina en la posición $(1, 2)$, otra en la posición $(2, 4)$, otra en la posición $(3, 1)$, y la última en la posición $(4, 3)$. De esta forma, la representación hace uso del concepto de permutación de los números 1 al n, y con la propiedad que solamente podemos encontrar colisiones diagonales disminuyendo así fuertemente el espacio solución. La cantidad de permutaciones de los números 1 al n crece exponencialmente con el tamaño de n, pero lo sorprendente es que la cantidad de soluciones, esto es, permutaciones con 0 colisiones, también crece a un ritmo exponencial aunque obviamente menor que la cantidad de permutaciones, como se ilustró en la introducción.

Estamos interesados en diseñar algoritmos de búsqueda local para N-Q. Un elemento fundamental en búsqueda local es la definición de un esquema de vecindades ya que el algoritmo se basa en explorar el espacio de soluciones siguiendo una trayectoria definida por puntos obtenidos en una vecindad de la solución actual. El algoritmo se detiene cuando no es posible encontrar en la vecindad del punto actual un elemento de mejor calidad.

Para el problema N-Q varios esquemas de vecindades se podrían establecer. En general, existe un compromiso en la definición del tamaño de la vecindad dado que vecindades de gran tamaño pueden generar puntos de buena calidad pero a un costo computacional alto; recíprocamente, vecindades reducidas serán examinadas rápidamente pero podría ser difícil encontrar mejores soluciones; de esta forma, normalmente se eligen vecindades que no consuman demasiado tiempo en la búsqueda pero que generen buenas expectativas de encontrar soluciones mejores.

Nuestra proposición de vecindad es muy simple. Dada una n permutación, se trata de escoger aleatoriamente dos elementos distintos del n-vector e intercambiarlos; esto es equivalente a intercambiar dos columnas en la disposición de un tablero dado. Por ejemplo, para $n = 5$, un tablero particular lo representamos por $(2, 3, 1, 4, 5)$. Si intercambiamos las columnas 2 y 4 obtenemos el vector $(2, 4, 1, 3, 5)$. La figura 2 ilustra esta operación. Notamos que, en este caso, el vector resultante es una solución al problema de las 5 reinas.

En búsqueda local, necesitamos evaluar el costo de las soluciones. Este costo estará dado por el número de colisiones entre reinas en un tablero particular. En el ejemplo anterior, el vector $(2, 3, 1, 4, 5)$ tiene colisiones, en particular la reina en la columna 1 colisiona con la reina en la columna 2 y la reina en la columna 4 colisiona con aquella en la columna 5, y luego existen dos colisiones. Luego, en general, dado un n-vector interesa calcular en forma rápida el número de colisiones que posee. Crawford [4] diseñó la siguiente estrategia, se construyen dos arreglos llamados positivo y negativo. Cada uno de estos arreglos tiene un tamaño de $2n - 1$ elementos, uno por cada diagonal ya sea positiva o negativa, y cada uno de estos elementos contabiliza, para un tablero dado, el número de reinas que se encuentra en esa diagonal. Cada vez que una reina se posiciona en la fila i y columna σ_i, un contador se incrementa en la posición $n + i - \sigma_i$ del arreglo negativo para contabilizar esta reina. Análogamente, para la posición $i + \sigma_i - 1$ del arreglo positivo. Si $\pi = \langle \sigma_1, ..., \sigma_n \rangle$ es una permutación denotamos por π^- al arreglo negativo y por π^+ al arreglo positivo, de modo que el número total de colisiones está dado por:

$c(\pi) = \sum_{i=1}^{2n-1} [(\pi_i^+ - 1) + (\pi_i^- - 1)]$, donde se considera para las sumas solamente aquellos valores $\pi_i > 1$.

Esta forma de evaluación del número de colisiones asociadas a una permutación se puede implementar en tiempo $O(n)$ [4].

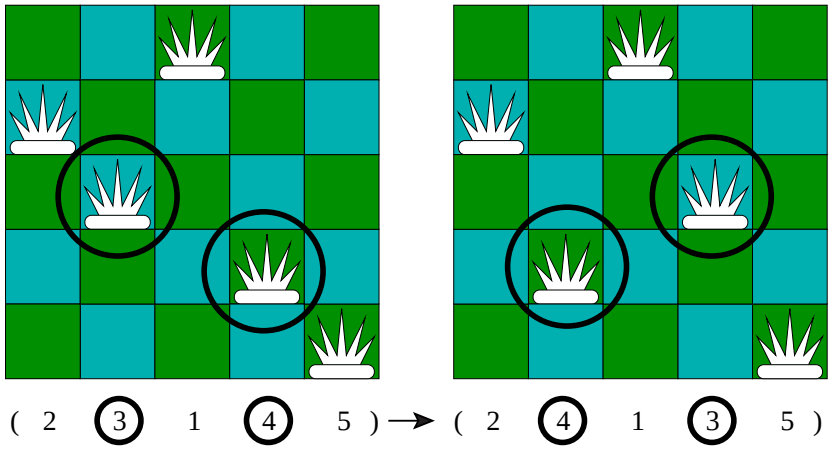


Figura 2: Ejemplo de vecindad para n=5

Notamos que diferentes permutaciones pueden tener diferente costo asociado, luego el objetivo de resolver el problema N-Q se traduce, entonces, en encontrar un valor de π tal que $c(\pi) = 0$. Es decir, una solución a N-Q es cualquier permutación de costo nulo. Es interesante comentar que N-Q, con esta formulación, es un problema de optimización discreta con valor óptimo conocido y tal que existe un número de soluciones que crece exponencialmente con n, para un n dado.

Existen varios algoritmos modernos que utilizan búsqueda local tales como simulated annealing, algoritmos genéticos o tabu search y ya se comentó en la introducción de su aplicación a N-Q. Sin embargo, discutiremos a continuación una heurística de búsqueda local simple.

3. BUSQUEDA LOCAL SIMPLE PARA N-Q

Búsqueda Local (BL) ha demostrado, durante varias décadas, su poder como un algoritmo muy simple y rápido que permite resolver, en forma aproximada, algunos problemas de optimización de alta complejidad. Uno de los casos más importantes es su aplicación al problema del Vendedor Viajero, donde heurísticas basadas en BL han logrado resolver instancias con miles de ciudades. Adicionalmente, este algoritmo también sirve como base para el diseño de metaheurísticas que en los últimos veinte años han provocado un impacto mayor en la resolución de problemas complejos, ver Glover y Kochenberger [8], para una serie de artículos sobre aspectos teóricos y prácticos ligados a BL.

El algoritmo BL parte de un punto inicial, típicamente generado aleatoriamente, y recorre parte del espacio solución generando en cada iteración un vecino del punto actual que tenga un costo menor (*mejoría iterativa*) o bien explorando completamente la vecindad de modo de alcanzar el vecino de menor costo (*máximo descenso*). El algoritmo se detiene cuando, a partir del punto actual no es posible encontrar, en su vecindad, un punto de costo menor. La salida del algoritmo es un punto factible cuyo costo representa una cota superior (para un problema de mínimo) para el valor óptimo del problema. BL tiene algunas variantes que permiten eventualmente mejorar su desempeño tales como partidas múltiples y considerar varias vecindades.

Para N-Q, BL partirá con una permutación aleatoria y repetidamente intentará encontrar un vecino mejor para la solución actual. Un punto crucial entonces será definir la vecindad. Una forma simple consiste en intercambiar dos elementos elegidos al azar del vector permutación, lo cual podría resultar en una disminución del número de colisiones. Dado que esto podría no ocurrir, necesitamos seguir buscando dentro de la vecindad hasta encontrar una permutación mejor. En el peor caso, podríamos explorar completamente la vecindad y no tener éxito; en este caso, la búsqueda termina y da como salida la permutación actual. Notamos que si buscamos completamente la vecindad entonces el número de búsquedas por un vecino mejor es igual al número de pares diferentes de componentes que podemos elegir en la n-permutación, o sea el tamaño de la vecindad es $\binom{n}{2} = O(n^2)$. Nuestra implementación considera un máximo de n^2 intentos notando que, dado el caracter aleatorio de las elecciones, es posible que existan repeticiones en la búsqueda de un vecino mejor. Luego, la regla de detención será la detección de una permutación con costo nulo, o bien alcanzar el máximo número de intentos n^2. La figura 3 ilustra el algoritmo de búsqueda local propuesto (BLNQ).

Con el objetivo de acelerar la búsqueda, también experimentamos con L =Máximo número de intentos igual a $O(n)$, en particular, $L = 3n$. La tabla 2 muestra los resultados obtenidos por el algoritmo BLNQ, en términos de colisiones promedio sobre 100 ejecuciones, y considerando $L = n^2$ y $L = 3n$.

```
PROCEDURE BLQ()
BEGIN
intentos :=  0;
L := n^2;
x := permutacion(n);
cx := costo(x);
IF (cx=0) THEN
        RETURN;
WHILE(b=false)
        BEGIN
        y := vecino(x);
        cy := costo(y);
        IF (cy < cx) THEN
                BEGIN
                x := y;
                cx := cy;
                IF(cx=0) THEN
                        b:=true;
                ELSE
                        intentos := 0;
                END
        ELSE
                BEGIN
                intentos := intentos + 1;
                IF (intentos > L OR cx = 0) THEN
                        b := true;
                END
        END
END
```

Figura 3: algoritmo BLNQ

n	$L = n^2$	$L = 3n$
10	0,97	1,46
20	1,27	2,39
30	1,14	3,19
40	1,06	4,06
50	0,97	4,47
60	0,9	5,35
70	1	5,54
80	0,77	6,14
90	0,93	6,37
100	0,91	7,24
200	0,31	10,04
300	0,17	12,72
400	0,05	14,22
500	0,02	16,38
600	0,03	17,08
700	0,01	18,74
800	0	18,81
900	0	20,31
1000	0	20,73
2000	0,00	26,5
3000	0,00	29,4
4000	0,00	31,6
5000	0,00	33,0
6000	0,00	32,2

Tabla 2: Número promedio de colisiones tras 100 ejecuciones, considerando $L = n^2$ y $L = 3n$

Además probamos con factores mayores que 3n, en particular $L = 10n$, pero el costo final no mejoró sustancialmente. Para $L = 3n$, notamos que BLNQ no funciona bien aumentando el número de colisiones con el tamaño de n. Sin embargo, para $L = n^2$ BL converge rápidamente a encontrar soluciones y, a partir de $n = 700$, las 100 ejecuciones obtienen una solución.

También realizamos experimentos con BL pero usando una representación matricial y con una vecindad de tamaño mayor, donde dada una configuración de reinas en el tablero obtenemos un vecino simplemente cambiando una reina a otra posición desocupada dentro del tablero.

Los resultados obtenidos fueron muy pobres dando origen a un número creciente de colisiones con el tamaño de n.

BLNQ usando $L = n^2$ se ve muy robusto en relación a su desempeño en función del tamaño de la entrada n. La tabla 3 muestra información adicional relacionada con el desempeño del algoritmo. En particular, para cada n, se muestran los resultados promedios (sobre 100 corridas) para el costo inicial, el costo final y la cantidad de mejorías alcanzadas en la búsqueda. Los resultados verifican la idea que los cambios locales permiten disminuir levemente el costo y, en promedio, se tiene poco más de una unidad en el costo por mejoría lograda.

Diversos trabajos se han realizado implementando algoritmos genéticos [4, 9, 10]. Sin embargo, a pesar de probar con una serie de operadores, resulta muy dificil de encontrar soluciones en tiempos razonables. Se pueden generar tableros con pocas colisiones muy rápidamente, pero llegar a una solución se puede tornar un proceso lento.

Realizamos experimentos con algoritmos genéticos probando diversas variaciones tales como operadores de cruzamiento, mutación, tamaño de la población, obteniendo resultados muy similares a los mencionados por Crawford [4] y ratificados por Hynek [10], respecto al hecho que el cruzamiento entre dos tableros no necesariamente generan un hijo de mejor calidad, obligando a reducir las potencialidades de un algoritmo genético a la capacidad de la componente mutación.

Esto último puede justificar, en parte, que la estrategia de algoritmos genéticos sea superada por una estrategia más simple de BL.

El desempeño de BLNQ hace recordar el algoritmo de Sosic y Gu (SG) [17], un algoritmo probabilístico muy rápido, que privilegia una fase de construcción de una solución, y posteriormente, una fase de reparación de la solución parcial encontrada en la primera fase. En términos de la obtención de soluciones, los dos algoritmos, SG y BLNQ tienen un desempeño similar en el sentido que, a partir de un cierto valor de n (n=700), alcanzan una solución con posibilidad muy cercana a 1.

n	Costo Inicial	Vecinos Analizados	Mejoras Realizadas	Costo Final
10	5,3	140,4	3,5	0.97
20	10,2	475,6	7,6	1.27
30	14,4	1099,1	10,7	1.14
40	21,6	1863,8	17,5	1.06
50	26,5	3103,2	21,9	0.97
100	55,6	12098,5	45,2	0.91
200	104,4	35430,1	88,7	0.31
300	158,2	39866,8	130,5	0.17
400	212,3	86344,7	177,4	0.05
500	262,7	94892,2	218,9	0.02
600	307,5	167051,5	259,3	0.03
700	367,9	120751	312,3	0.01
800	424,5	148108,4	353,46	0
900	476,4	171108,5	399,7	0
1000	530,5	172121,3	444,8	0

Tabla 3: Valor promedio para los distintos valores de N usando BL, considerando 100 corridas y $L = N^2$

Adicionalmente, estudiamos el problema de obtener un conjunto de soluciones diferentes para N-Q. Realizamos experimentos con SG y BLNQ concluyendo que cuando estos algoritmos alcanzan soluciones, todas éstas son diferentes. Esta diversidad alcanzada es muy interesante ya que no es posible observarla con los algoritmos explícitos comentados en la sección 2.

n	50	100	200	300	400	500	600	700	800	900	1000	2000
Soluciones	0	6	13	19	20	19	20	20	20	20	20	20

Tabla 4: Soluciones encontradas para el algoritmo de Gu después de 20 pruebas para cada valor de n

4. CONCLUSIONES

Hemos diseñado un algoritmo de búsqueda local para N-Q, el primero de este tipo según nuestro conocimiento. El algoritmo tiene complejidad computacional exponencial en el peor caso, como es típico para esta clase de algoritmos, pero tiene tiempo de corrida observado $C = n^3$, donde C es una constante aproximada al valor 1/200,000. Del punto de vista de la calidad de la solución, el desempeño del algoritmo es claramente superior al de algoritmos genéticos propuestos, y podría ser aún mejorado tanto en tiempo de ejecución (por ejemplo, calculando el costo en tiempo constante), como en el aumento de la probabilidad de obtención de soluciones para valores de n, $n \leq 700$ (por ejemplo, haciendo L depender de n).

5. AGRADECIMIENTOS

Los autores agradecen los comentarios de los revisores y, en particular, el comentario referido a la complejidad computacional del algoritmo propuesto.

Referencias

[1] B. Abramson and M. Yung. Divide and conquer under global constraints: A solution to the n-queens problems. *J. Parallel Distrib. computing*, 6:649–662, 1989.

[2] Anonymous. Unknown. *Berliner Schachgesellschaft*, 3:363, 1848.

[3] B. Bernhardsson. Explicit solution to the n-queens problems for all n. *ACM SIGART Bulletin*, 2:7, 1991.

[4] K.D. Crawford. Solving the n-queens problem using genetic algorithms. In *Proceedings ACM/SIGAPP Symposium on Applied Computing, Kansas City*, pages 1039–1047, 1992.

[5] A.E. Eiben, P.-E. Raué, and Zs. Ruttkay. Solving constraint satisfaction problems using genetic algorithms. In *Proceedings of the 1st IEEE World Conference on Computational Intelligence*, pages 542–547. IEEE Service Center, 1994.

[6] L.R. Foulds and D.G. Johnson. An application of graph theory and integer programming: Chessboard nonattacking puzzles. *Mathematical Magazine*, 57:95–104, 1984.

[7] J. W. L. Glaisher. *Philosophical Magazine*, 18:457–467, 1874.

[8] F. Glover and G. Kochenberger. *Handbook of Metaheuristics*. Kluwer, 2002.

[9] Abdollah Homaifar, Joseph Turner, and Samia Ali. The n-queens problem and genetic algorithms. In *Proceedings IEEE Southeast Conference, Volume 1*, pages 262–267, 1992.

[10] J. Hynek. The n-queens problem revisited. In *Proceedings of the ICSC, Symposia on Intelligent Systems and Applications ISA'2000*. ICSC Academic Press, 2000.

[11] A.N. Souza I.N. da Silva and M.E. Bordon. A modified hopfield model for solving the n-queen problem. pages 509–514. IJCNN'00, 2000.

[12] S. Lee and J. Park. Dual-mode dynamics neural networks for combinatorial optimization, 1994.

[13] L.V.Kalé. An almost perfect heuristic for the n nonattacking queens problem. *Information Processing Letters*, 34:173–178, 1990.

[14] J. Mańdziuk and B. Macukow. A neural network designed to solve the n-queens problem. *Biological Cybernetics*, 66:375–379, 1992.

[15] Franz Nauck. Schach. *Illustrierter Zeitung*, 361:352, 1850.

[16] I. Rivin, I. Vardi, and P. Zimmermann. The n-queens problem. *The American Mathematical Monthly*, 101:629–639, 1994.

[17] Rok Sosic and Jun Gu. Efficient local search with conflict minimization: A case study of the n-queens problem. *IEEE Transaction on Knowledge and Data Engineering*, 6(5):661–668, 1994.

[18] H.S. Stone and J.M. Stone. Efficient search techniques - an empirical study of the n-queens problem. *IBM J. Res. Develop.*, 30(3):242–258, 1986.

Treating Components and Connectors Explicitly during Software Design

An Approach Based on Software Architecture

Marco Antônio Fagundes de Moraes
Universidade Federal do Pará, Departamento de Informática, Belém, Brazil, 66075-110
Tribunal Regional Eleitoral do Pará, Secretaria de Informática, Belém, Brazil, 66015-902
mfagunde@tre-pa.gov.br

and

Alexandre Marcos Lins de Vasconcelos
Centro de Informática – Universidade Federal de Pernambuco,
Cidade Universitária, Recife, Brazil, 50732-970
amlv@cin.ufpe.br

Abstract

Software Architecture(SA) is considered a critical factor in software design. The adoption of an approach that treats architecture explicitly, emphasizing the separation between "computation" and "communication", is considered an important aspect in obtaining certain benefits (e.g., reuse in high levels of abstraction). However, explicit treatment of SA has not been the focus of the most used software processes, due to some reasons: SA use specific terminology (components, connectors and configuration); the fact that SA is an emerging discipline; and little support from available tools. In this paper, we present ***ArcADe*** (software **Arc**hitecture-based **A**nalysis and **De**sign process), a process that integrates concepts and patterns largely used in SA. This process has been influenced by the RUP (Rational Unified Process) and deals with relationships between requirements and architectural abstractions, elaboration, representation and materialization of software architecture.

Keywords: Software Architecture, Software Reuse, Rational Unified Process (RUP).

1 Introduction

Software architecture is considered a critical factor in software design. The use of an approach that adequately addresses architecture (e.g., Shaw and Garlan [15]), emphasizing the separation between computation (components) and communication (connectors), can be considered a crucial aspect to obtain certain benefits. Some of these benefits include reuse in high levels of abstraction, development management and software evolution facilities, and potential application of other approaches that promote reuse (e.g., Design Patterns[6], and Middleware[4]).

Some approaches to architecture adopted in widely used software processes (e.g. Rational Unified Software Process - RUP [9]) do not clarify the separation between computation and communication in high abstraction levels. In this context, it is visible that there is no natural integration between the SA discipline and widely used processes. Some reasons explain this lack of integration: SA is a recent discipline; SA is not supported by most of the available tools; SA is described through components, connectors and configuration, and not only through class diagrams, as it is common in several object-oriented projects.

To solve the problem of the lack of integration between SA and the broadly used processes, some approaches have been proposed, for example: use UML (Unified Modeling Language) as ADL (Architecture Description Language) [7], Catalysis [3], and MDA (Model-Driven Architecture) [10]. The first approach prescribes the representation of the architecture through UML, instead of traditional ADLs, that present a certain formal rigidity and complex syntaxes. The process Catalysis encompasses all the development cycle, deals with components and connectors explicitly, and uses UML to construct its models. MDA uses UML and emphasizes the construction of models that don't depend on platform and implementation technology, with subsequent translation of these models to platform-specific software architectures.

Considering the aforementioned approaches, two points should be highlighted. First, the approaches that use UML to describe the architecture not encompass all aspects of architecture design (e.g. refinement). Second, Catalysis and MDA are not yet broadly known in software development commercial environments.

Based on the context described, we identify the need of a process that could conduct the architectural design, dealing with components and connectors separately. At the same time, this process should use methods, techniques, and standards (e.g., UML) widely adopted in industry and academy. Because of these factors, we elaborated the ***ArcADe*** Model (Software **Arc**hitecture-Based **A**nalysis and **De**sign Process), which adopts the concepts and principles of SA as its base, and uses RUP elements to organize and represent its workflow. Due us model to adopt largely used patterns (e.g. UML), we believe that their acceptance will be facilitated.

Besides this introductory section, this paper is organized as following: Section 2 introduces the basic concepts used in our proposal. Section 3 presents the structure of ***ArcADe***. Section 4 presents a case study. Finally, Section 5 presents an analysis of the model, conclusions, and possible future work.

2 Motivation

The problem of lack of integration between SA and a widely used process, specifically the RUP, was initially treated considering two main aspects: SA approach and how the RUP deals with architecture. These aspects are describes in this section.

2.1 Software Architecture

There are several definitions of SA (e.g. [2],[15]), among which Shaw & Garlan's definition [15] is the most traditionally adopted by the SA community. This definition, used as the basis of the proposed model, asserts that software architectures are generally described in terms of three basic abstractions: components, connectors and configuration (as depicted in Figure 1.a). Essentially, SA presents a software description in which the "computation" (included in components) and the "communication" (embedded in the notion of connectors) are clearly separate [14]. This separation reduces the coupling among the parts of a system, increasing the opportunities of reuse of components and connectors in other contexts.

2.2 Treatment Architecture in the Rational Unified Process (RUP)

RUP [9] is a generic software development process developed by Rational Software Corporation. RUP has four main characteristics: driven by use cases, centered in the architecture, and prescribes an iterative and incremental development. This process has five basic concepts: worker, artifact, activity, workflow, and workflow detail. Worker is a role that can be given to a person or group. Artifact is information produced, modified, or used. Activity is a unit of work that can be executed by a worker. Workflow is a sequence of activities that are grouped in stages of the development (e.g. requirement workflow). Workflow details are used to structure a flow in terms of activities, workers, input and output artifacts.

The RUP adopts a particular approach to deal with architecture (4+1 View Model [9] – see Figure 1.b). In this approach, the architecture description encompasses five views. Architects capture their design decisions in four views (logic, process, implementation, and deployment), and use the fifth view (use cases) to illustrate and validate the previous four.

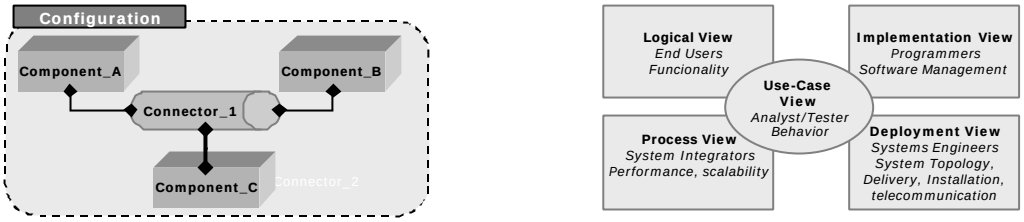


Figure 1: (a) Overview of AS (b) 4+1 View Model

To analyze these views (Figure 1.b), the architecture is described as a collection of use case, design, and deployment models. This consideration is not in conformity with the definition presented in subsection 2.1, in respect to the terminology used in SA (components, connectors and configuration) and in the little emphasis given to the reduction of coupling, consequently reducing reuse opportunities.

Such aspect hinders an adaptation/extension of RUP to treat SA, because the difference is in the base concept: "architecture." Thus, a probable adaptation of RUP would imply in the redefinition/reorganization of all the activities involved in the treatment of the architecture. However, the benefits of RUP, such as, diffusion of use and tool support should not be discarded. Based on this context, we have formulated a process model that integrates SA with RUP elements. This model, named ***ArcADe*** [12] (Software **Arc**hitecture-Based **A**nalysis and **De**sign Process), adopts the concepts and principles of SA as its base, and uses RUP elements to organize and represent its workflow.

2.3 Scope and Use of Model

The *ArcADe* can be applied to any software development organization wanting to reach very abstract levels of reuse and to improve its software development, evolution, operation, and support capabilities. The model doesn't assume a particular organizational structure or management philosophy. On the other hand, it assumes an iterative and incremental development model, organizing the analysis and design activities, so as to assist the professionals involved in the understanding and use of the SA discipline during software development. Concerning the use of *ArcADe*, we highlight the aspects listed in Figure 2.

Who?	Why?	How?	When?
Analysts, Designers, and Software Architects.	To obtain the benefits of SA (e.g., clear structuring of software, reuse in a high level of abstraction, easiness of complexity management through decomposition of the problem.).	Considering a scenario of use (see Section 4), following the workflow defined in the model and executing its activities according to its guidelines.	During the mapping of the requirements to the architecture, together with the realization of the architectural abstractions, in order to obtain a detailed design to be submitted to the implementation.

Figure 2: Use of *ArcADe*

ArcADe can be used in big and complex software projects (e.g. systems with different platforms, multiple programming languages), as well as projects of reduced size and complexity. Additionally, the model can be integrated to an existing process in the organization, as long as the referred process recognizes the input and output artifacts (information produced, modified, or used) of the model. In the next section, we present the abstraction levels and the organization of the proposed model.

3 The *ArcADe* Model

ArcADe (software **Arc**hitecture-based **A**nalysis and **De**sign process) is an analysis and design process model that integrates SA with RUP elements (concepts, methods, and techniques). The model adopts SA as the basis for definition of the stages of development and uses RUP elements to organize and to represent the process in a workflow. We emphasize that our model is not an adaptation/extension of RUP, but an integration of SA and the elements of that process.

It is important to emphasize that the integration must encompass the functional requirements (FRs) as well as the non-functional (NFRs) ones. However, in our proposal, we considered only the functional requirements, because RUP is driven by use cases, which are only employed in the description of the functional requirements.

A future adaptation of the proposed model to treat non-functional requirements can be made using techniques proposed in [14], for example. In the following subsections, we present the abstraction levels, organization and the structure of the proposed model.

3.1 Abstraction Levels

ArcADe follows the iterative and incremental development model adopted by the RUP. In this model, the basic idea is to compose an abstract architecture from requirement analysis, supplying a description of components and their interactions in a high level of abstraction. This architecture will have a subset of the architectural abstractions (see architecture design dimension) mapped to a concrete architecture, supplying an architectural representation more directed to the implementation (see detailed design dimension). In the level of detailed design, the architectural abstractions (components and connectors) will be materialized through the use of project strategies (e.g. Design Patterns) or through the incorporation of existing products (e.g. Commercial Off-The-Shelf – COTS [1], Middleware CORBA [16]). This process is repeated (iteratively) until the global architecture (incremental) has been materialized in terms of design elements. This scenery is presented in the Figure 3.

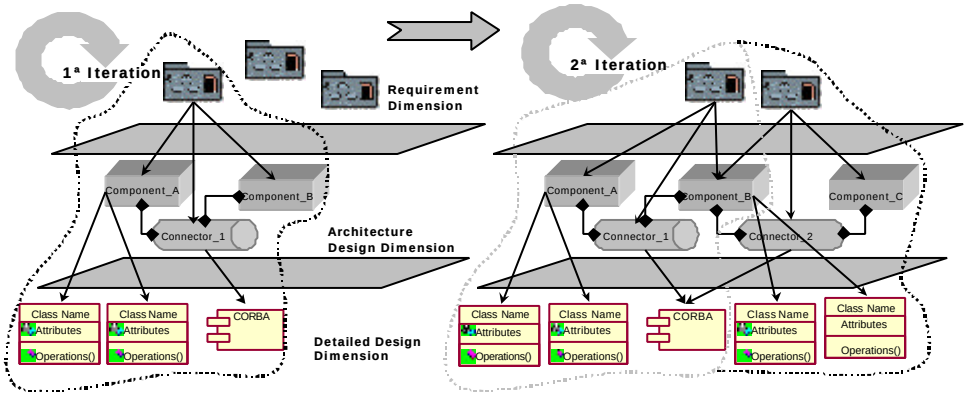


Figure 3: Iterative and Incremental Development in *ArcADe*

3.2 Organization of the Model

The RUP specifies activities, artifacts, and guidelines to treat architecture (according to its approach – see Subsection 2.2). These elements are in the Analysis and Design workflow. Considering this workflow and the abstraction levels presented in subsection 3.1, we have structured *ArcADe* in eight workflow details (see Figure 4.b).

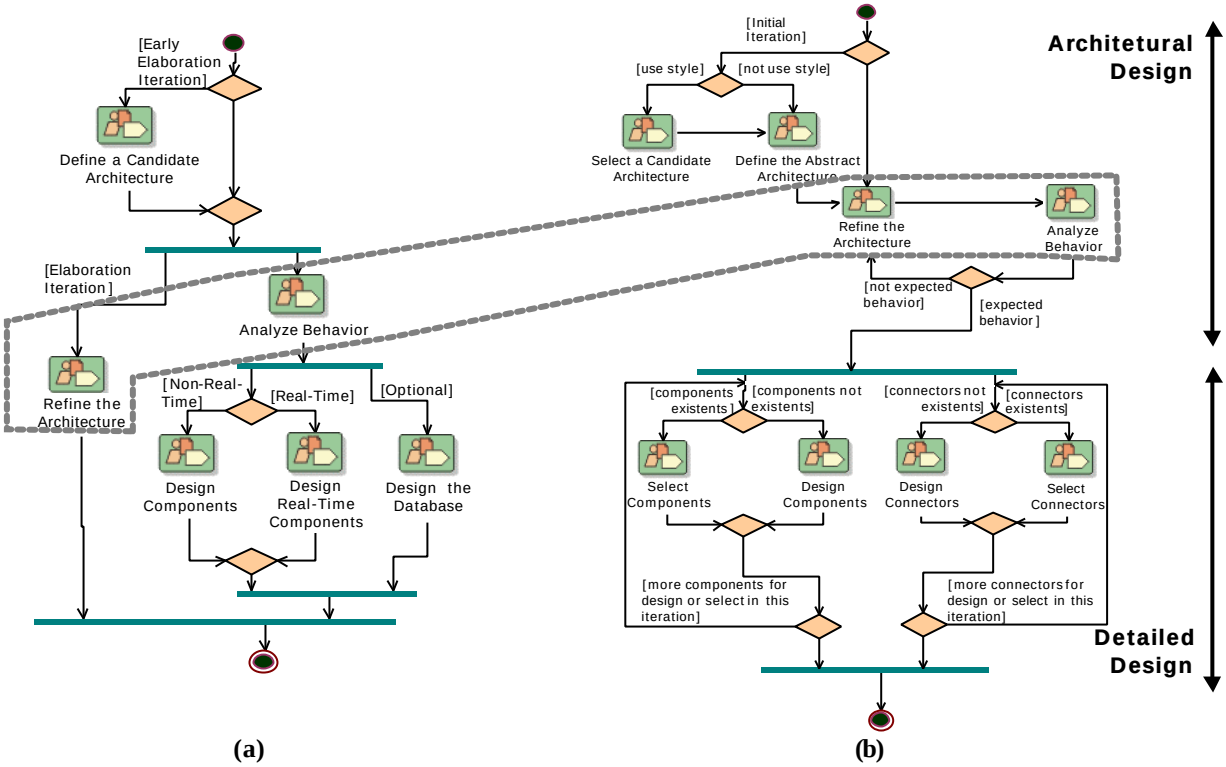


Figure 4: (a) RUP Analysis and Design workflow (b) *ArcADe* Model

The activities presented in the diagrams of Figure 4 represent "workflow details" (see RUP – Figure 4.a – and *ArcADe* – Figure 4.b) that, in certain cases, have the same name. However, in the proposed model, the activities and guidelines involved are based on SA definition (see Subsection 0). Considering the differences in the organization of the two diagrams, it is important to emphasize the execution of the workflow details Refine the Architecture and Analyze Behavior (see highlighted area in Figure 4). In the RUP, this execution is in parallel, but in *ArcADe* it is sequential. We have chosen this organization because the refinement may cause structural changes in the architecture (components and connectors, can be decomposed, aggregated, or removed). Because of this structural change, it is necessary to make an analysis of behavior to verify whether the structural properties are preserved after the refinement. For example, if two components don't communicate in the abstract architecture, they should not be linked in the refined architecture.

3.3 The *ArcAde* Workflow Details

The ***ArcAde*** supplies a compilation of a set of approaches based on SA (e.g. [4], [5] and [7]) organized in eight workflow details. Each workflow is described as a set of activities, which are detailed step by step. The model also encompasses strategies for software reuse (e.g. Design Patterns and COTS). All these elements are gathered with the objective of guiding developers through the requirement-architecture-implementation abstraction levels.

In the next subsections, we present a summarized description of each workflow detail of the proposed model. This description is grouped according to the levels of abstraction presented in subsection 3.1. We emphasize that some activities/steps of the workflow details were elaborated from the approaches based on SA (as mentioned in the beginning of this section), while others were imported from the RUP.

To illustrate this situation, the Figure 5 and Figure 6 show the workflow details, activities and steps of ***ArcAde***, using a convention in the activity/step to indicate its origin: (*) adapted; (+) elaborated; (RUP) imported from the RUP; and (no indication) imported in its original form from the used approach.

3.3.1 Architecture Design Dimension

This level assembles activities designed to relate requirements with an abstract architecture, which will be refined through aggregation, decomposition, or removal of components and connectors. This refinement causes structural changes. Thus, it is important to guarantee the preservation of the architecture's properties by analyzing its behavior. In this level, four workflow details are emphasized: Selection of candidate architecture; Definition of the abstract architecture; Refinement of the architecture; and Behavior analysis.

Select candidate architecture. The CBSP (Component-Bus-System-Property) approach [5] is used to relate domain elements with architectural abstractions (components and connectors) in order to elaborate an intermediary model of the architecture. This model must be related with architectural styles [15] for selection of a candidate architecture.

Define the abstract architecture. Components and connectors of the intermediary model (produced in the previous workflow detail) are mapped to the vocabulary of the candidate architecture and organized according to the configuration rules of an architectural style. As a result, a preliminary model of the architecture is produced. This model shows the services provided by the components and must be converted to the definitive representation of the software architecture using UML. In an adaptation of the proposed model with the objective of dealing with non-functional requirements (as mentioned in the beginning of section 3), it would be possible to use methods to analyze the architecture, such as Software Architecture Analysis Method (SAAM) [2] (like ***ArcAde***, SAAM is based in scenarios). However, this topic will be left for future investigation.

Refine the architecture. The abstract architecture (produced in the previous workflow detail) is submitted to refinement rules [13] in order to obtain an architecture in a less abstract level, which will be the basis of the materialization phase of the architectural abstractions.

Analyze behavior. Due to the fact that UML does not support rigorous behavior verification, as the ones allowed by Wright [11], we have performed a limited analysis through the construction of state diagrams for the components (abstract and concrete) involved in the refinement. After analyzing the aforementioned resulting diagrams, it could be verified if the behavior of the global architecture been preserved after the refinement.

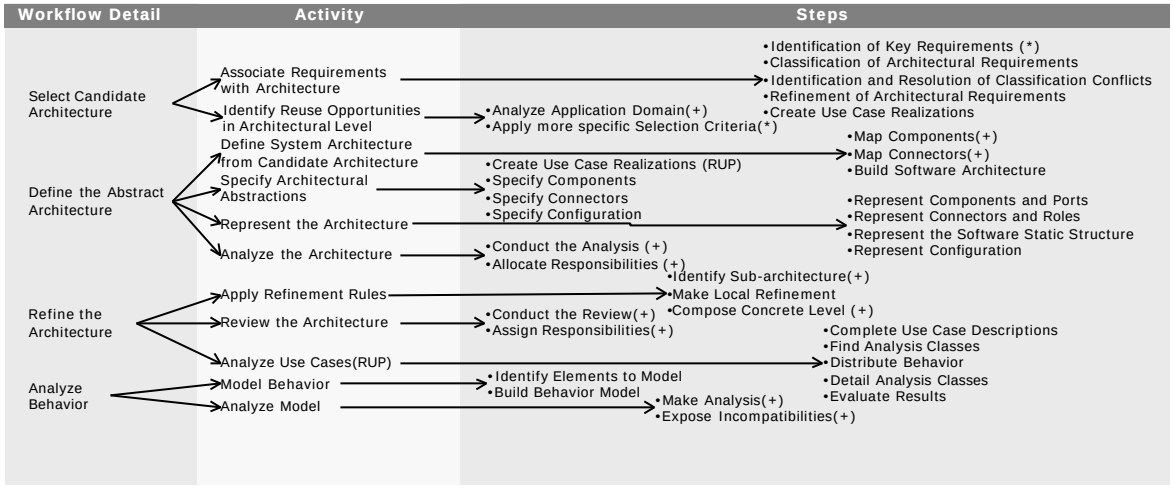


Figure 5: Activity and Step of the Architecture Design Dimension

3.3.2 Detailed Design Dimension

This level assembles activities designed to provide a concrete architecture, which supplies a more implementation-oriented representation, including design strategies (e.g. Design Patterns) or selection of available components and connectors (e.g. Commercial Off-The-Shelf [1]). In this level, four workflow details are emphasized: design or selection of components and design or selection of connectors.

Design components. In a more concrete design level, components can be built through the application of Design Patterns. According to this approach [6], the patterns that supply the solution to the problem are selected. Next, it is necessary to detail class properties (e.g. attributes) and use cases involved, in order to define the structure and functionality of the component. Finally, the OMG IDL (Interface Description Language) [16] is used to specify the interface of the component.

Select components. When confronted with other processes (e.g. RUP), an advantage of ***ArcADe*** is that it guides the construction (design) and also supports the incorporation of existing products (e.g. COTS). For the incorporation, we use the CRE method [1], which identifies the candidate products that fulfill a selection criterion (e.g. Component Interface). This method indicates, through a rank, which product should be integrated in the final system. In case no product is selected, we suggest the execution of the workflow detail Design Components.

Design connectors. Sometimes, the basic type of connections (e.g. Remote Procedure Call) are not enough to fulfill the communication needs between the components. In this case, we have suggested an extension of basic types, using the approach [17] in order to produce a new connector.

Select connectors. Similarly to component reuse, connectors can be mapped to middleware technologies (e.g. ORB). In order to guide the selection process of appropriate middleware, we again suggest the CRE method [1]. From the selection criterion (e.g. support to multiple platforms) the candidate technologies are identified. The CRE method indicates which of these technologies will be integrated in the final system. In case no technology is selected, we suggest the execution of the workflow detail Design Connectors.

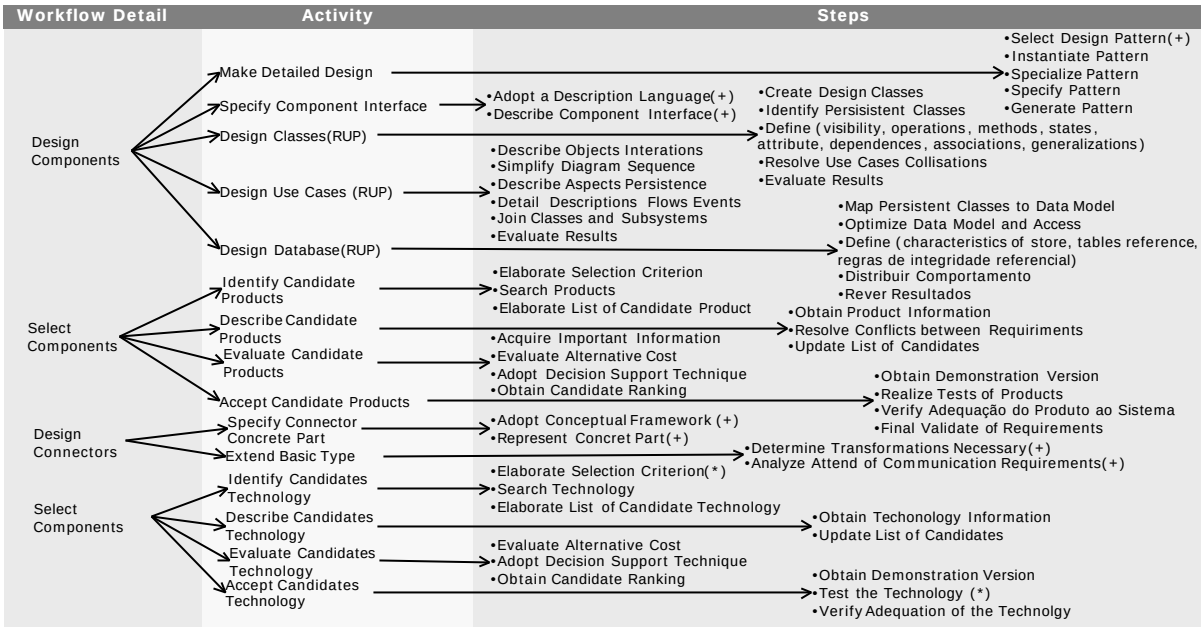


Figure 6: Activity and Step of the Detailed Design Dimension

4 A Use Scenario for *ArcADe*

This section illustrates the use of ***ArcADe*** in a real case study, in which the activities contained in the workflow details are executed based on their guidelines. In this study, we used an application, SIG@UFPE, developed in collaboration with the Information Technology Center of Federal University of Pernambuco (Núcleo de Tecnologia da Informação da Universidade Federal de Pernambuco – NTI/UFPE).

4.1 The SIG@UFPE Application

One of the goals of SIG@UFPE (Information and Academic Administration System of UFPE) is the automation of the academic control process in all teaching levels: under-graduation, graduation, and extension. It provides services like registration, accompaniment, and term finalization. All these operations are accomplished in a decentralized and secure way through the Web.

Because of the existence of several possible execution scenarios of ***ArcADe***, and because of the size of SIG@UFPE, we illustrate the use of the proposed model considering only two aspects. First, we selected a representative execution scenario, involving three elements: use of architectural styles; use of nonexistent components; and employment of middleware to materialize connectors. Second, we limited the scope of SIG@UFPE to some of its functionalities (see Subsection 4.2) and we executed each activity of the workflow details of the proposed model according to their guidelines.

This scenario implied in the execution of six workflow details: Select Candidate Architecture, Define the Abstract Architecture, Refine the Architecture, Design Components, and Select Connectors. This execution is detailed in the following subsections

4.2 Base Artifacts for the Execution of the Workflow Details

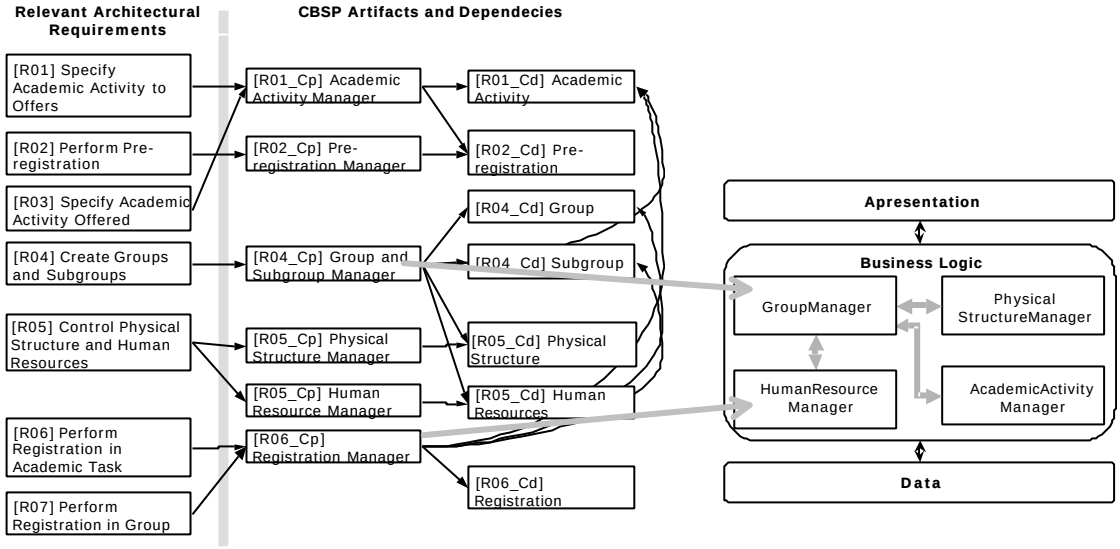
ArcADe considers the requirement document as one of its main inputs. In the SIG@UFPE system, the module used in the case study deals with registration planning. The main functional requirements of this module are showed [R01]...[R07] (see Figure 7.a). Among these, we have limited the scope of the study to the requirement [R04], because we have considered it one of the most representative requirements, due to its complexity degree, number of elements involved during its realization, and need of communication with other components to accomplish its responsibilities.

4.3 Execution of *ArcADe* Workflow Details

4.3.1 Workflow Detail: Select Candidate Architecture

Initially, we associated the requirements with the software architecture via CBSP (Component-Bus-System-Property) approach [5], in which the requirements ([R01]...[R07]) were considered of architectural relevance and classified as processing component (Cp). The dependencies among these components were obtained from the communication needs among them. For example, in Figure 7.a, the component Group and Subgroup Manager depends on information from Physical Structure (data component – Cd).

Components and dependencies help the architect to find an appropriate architectural style. Considering the application domain (Web) the client-server style [15] organized in three levels was selected (Figure 7.b). From the CBSP model (Figure 7.a) and this style, the abstract architecture of the system was defined.



(a) **(b)**

Figure 7: (a) CBSP Model (b) Service relationships

4.3.2 Workflow Detail: Define the Abstract Architecture.

As mentioned in the beginning of Subsection 4.2, we limited the scope of the case study to requirement [R04]. From this requirement, and considering the organization of the architecture in three levels: the GUI component is dedicated to the presentation level; the Manager components are allocated to the level of business logic; and the data components are dedicated to the data level. This preliminary model, which shows the list of services, is depicted in Figure 7.b.

Next, the architectural abstractions are documented in a form based in [11]: Component Interface (Offered Services – input and output ports, Requested Services), Type, and Semantics. The Figure 8 presents a summary of the specification of the GroupManager component using this form.

Interface

Offered Services

1. Input Ports
   specifyAcademicActivity: Offers the service responsible for the assignment of an academic activity to a group.
   ...
2. Output Ports
   listGroup: Supplies data related to a specific group.
   ...

Requested Services

listClassrooms: Requests the service responsible for obtaining the available rooms...

Type

GroupManager: This type of component encapsulates the functionalities to create groups...

Semantics

In order to create a group, the GroupManager must request the academic activities ...

Figure 8: GroupManager component specification (partial)

With the specification of the architectural elements (see The Figure 8) and with the service list model (see Fig. 7.a), we represented the architecture in UML using the approach described in [7] (see Fig. 7.b), where we defined the stereotypes *<<Arch-Component>>* and *<<Arch-Connector>>* to model components and connectors, respectively. The interfaces of the architectural elements were modeled through interface classes. The ports and roles were represented by the stereotypes *<<in>>* and *<<out>>*, indicating the direction of communication. The model of Figure 9.b will be submitted to refinement rules.

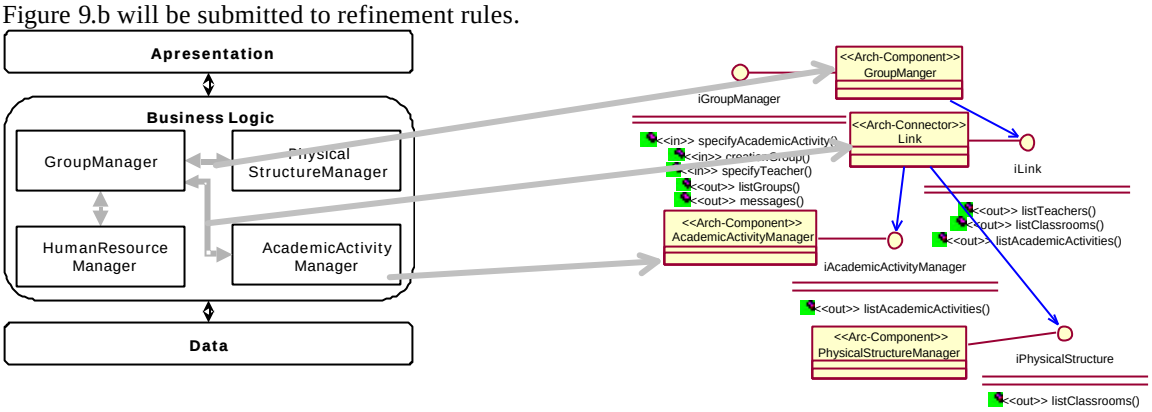


Figure 9: (a) Service relationships (b) UML Representation of the architecture

4.3.3 Workflow Detail: Refine the Architecture

With the abstract architecture defined, the next step is to translate it to a more concrete level. To illustrate this, we refined the component GroupManager by decomposing it in two (see Figure 10). GroupCreationManager offers the service of creating a group (assignment of the academic activity) and managing group details (e.g. classrooms, schedule, etc). TeacherAllocationManager offers the services for allocation of one or more teachers to a group previously created.

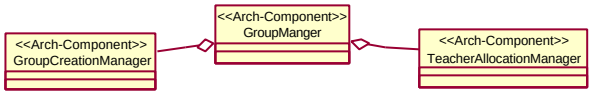


Figure 10: Component GroupManager Refinement

4.3.4 Workflow Detail: Analyze Behavior

Because the refinement may cause structural changes, we must verify if the refined architecture preserves the behavior of the abstract architecture. To accomplish this, we built an UML state diagram for each component involved in the refinement (see Figure 10). As mentioned in Subsection 3.3.1, UML supports a limited behavior analysis, in which we analyze the state diagrams and observe if the behavior of the components resulting from the refinement (see Fig. Figure 11.b and Figure 11.c) is contained in the behavior of the abstract component (see Figure 11.a). In the example, we verified that the behavior of the abstract architecture was preserved.

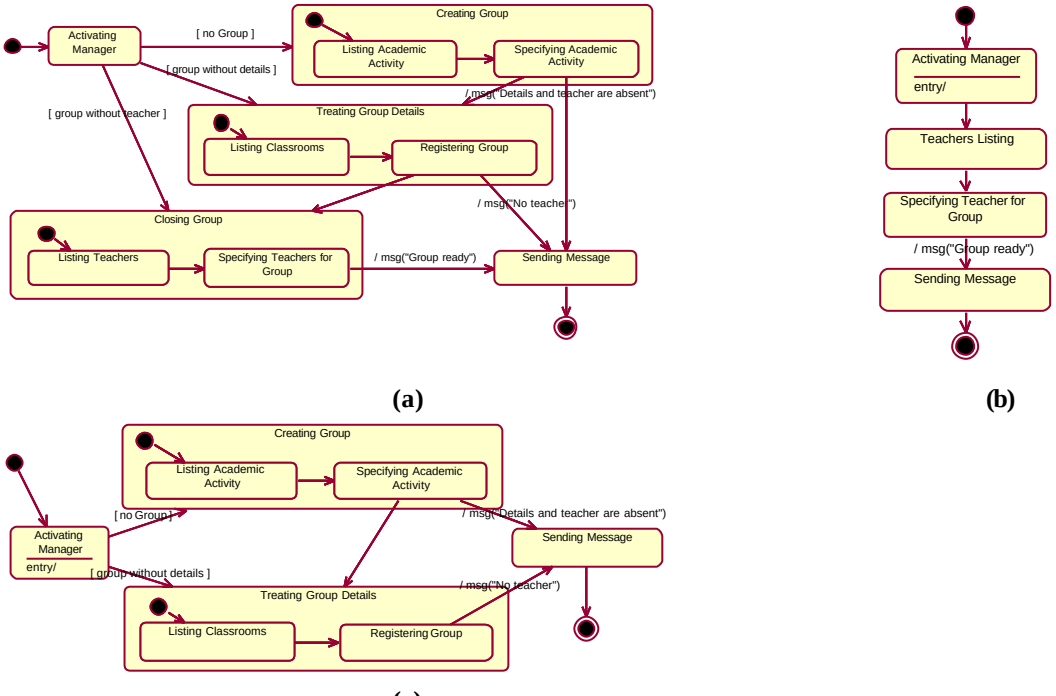


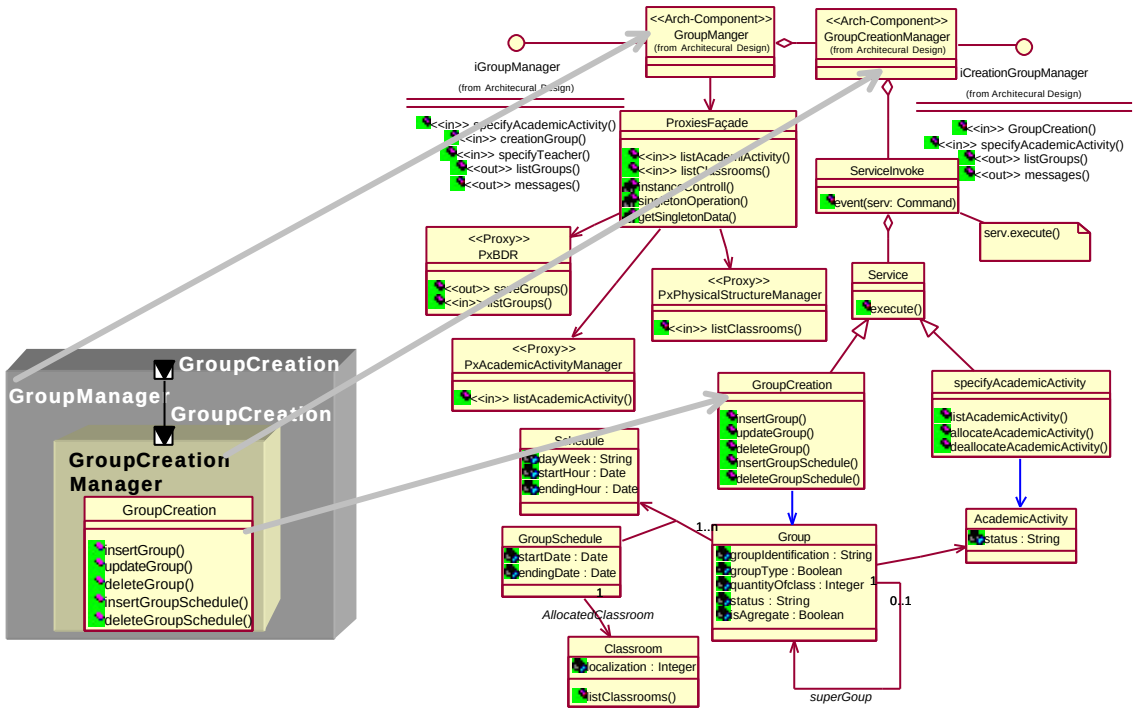
Figure 11: States Diagrams – (a) GroupManager (b)TeacherAlocationManager (c)GroupCreationManager

4.3.5 Workflow Detail: Design Components

Each component of the refined architecture must be implemented via design strategies. Observing the specification of the architectural component (see Figure 8) and the SA representation (see Figure 9.b), we selected three design patterns according to their purposes: Façade, Proxy, and Command. These patterns were applied in the design of the components GroupManager and GroupCreationManager. In this application, the patterns were detailed and instanced in the context of the components (see Figure 12).

In Figure 12.b, we can observe that the component was designed according to the SA discipline (Subsection 0), providing services through ports, which will be implemented by a set of operations. For example, in Figure 10.a the port "GroupCreation" of the external component "GroupManager" is linked to the port "GroupCreation" of the internal component that will be implemented by the class "GroupCreation". In other words, in the port "GroupCreation", the operations that will be provided are insertGroup(), updateGroup(), and so on.

After the design of the internal structure and functionality of the component, its interface must be specified with the OMG IDL (Interface Description Language).



(a) **(b)**

Figure 12: GroupManager Component - (a) Informal schema (b)UML detailed design

4.3.6 Workflow Detail: Select Connectors.

Initially, the options of middleware that are potential candidates are located according to the selection criterion (see Figure 13.a). After that, we selected the middleware that fulfilled the criterion (see Fig. Figure 13.b).

Selection Criterion
▪ Intra and Inter-process communication support
▪ Connector characteristics
▪ Platform support
▪ Communication Method

(a)

Technology	Developed by	Source
ILU	Xerox PARC	ftp://ftp.parc.xerox.com/pub/ilu/ilu.html
ORB(Visroker)	Inprise	http://www.inprise.com
RMI	Sun Microsystems	http://java.sun.com/products/jdk/rmi
RMI/IIOP	Sun Microsystems	http://java.sun.com/j2ee/

(b)

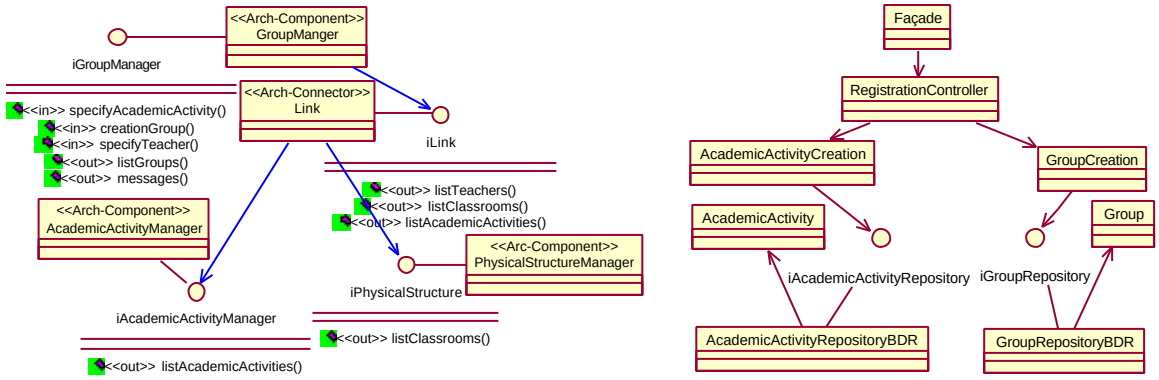
Figure 13: (a) Middleware selection criterion (b) List of candidate *middleware*

After that, we collected detailed information about the technologies, in order to expose the positive and negative characteristics of each one. Analyzing this information, we eliminated two technologies: RMI, for not promoting interoperability; and ILU, for not being so widely known, having little available documentation and support. Finally, we verified which technology better fulfilled the communication requirements. Then, we performed some tests to check legal and communication aspects. Considering the use of the decision-taking technique and the conformity with the tests, the middleware ORB (Visibroker) was selected.

Components and connectors must be integrated to compose the global architecture of the system. This integration should occur in later stages of the development process, particularly, in the system implementation stage.

4.4 Benefits Attained

The benefits attained with the use of ***ArcADe*** are the same ones supplied by SA (e.g. reuse in abstract levels and treatment of connectors as first-class entities). In contrast with the process used in NTI/UFPE (see Figure 14), we can highlight the aspect describe in this section.



(a) **(b)**

Figure 14: Architecture produced: (a) by ***ArcADe*** (b) by the process used in NTI/UFPE

First, in ***ArcADe*** the components are differentiated from each other through a stereotype indicative of its type. Second, the proposed model treats architectural elements in very abstract levels. For example, in the interface of the components, the services represent a set of operations to be provided. Particularly, the service "create groups" supplies the inclusion, exclusion and alteration operations for a group. Finally, in ***ArcADe***, connectors have first-class status, that is, the design of connectors becomes explicit (see class Link in Figure 14.a). As a result, there is a reduction in the coupling among system components, increasing the opportunities of reuse of components and connectors in other contexts.

NTI-RUP is based in RUP. As a result, it has the same differences in the treatment given to architecture, according to the definition used in our model (see Subsection 0 and 2.2). ***ArcADe***, on the other hand, is based on the SA discipline, resulting in its inherent benefits.

In NTI-RUP, in code level, each basic class (e.g., Group) generates a package (directory) that has the basic class, the repository interface, the collection class, and the RDB(Relational Data Base) repository class. In order to make it possible to reference a class in another package, it is necessary to import the package. As a result, the communication mechanisms are embedded in the components, making connector design not explicit.

In this scenario, we observed an increase in the coupling among components. Such an aspect reduces the potential reuse of components and connectors. On the other hand, ***ArcADe*** emphasizes the separation between computation elements and communication mechanisms, reducing coupling and facilitating component and connector reuse in other contexts. This is possible because our model treats component design and connector design explicitly and separately, resulting in language independence, treatment of legacy systems, and distribution facilities.

In the case study, we illustrated the implementation of a component (GroupManager) and the selection of middleware to implement a connector. It is worthwhile to point out that ***ArcADe*** considers the iterative and incremental model. Consequently, the other components and connectors must be implemented in future iterations, until the concrete level of the global architecture has been reached.

This case study was important for the incorporation of improvements in our model, because it made it possible to verify the coherence of the activities and steps involved in each workflow detail.

5 Conclusions and Future Research

This paper presented the ***ArcADe*** (Software **Arc**hitecture-Based **A**nalysis and **De**sign Process) model, which defines how the architecture must be explicitly treated in an iterative and incremental development process. The model treats computation elements (components) and communication mechanisms (connectors) explicitly and separately. This approach results in some benefits (e.g., reduction of coupling and consequent increase in reuse opportunities). This model has eight workflow details, which are grouped according to the abstraction levels of the model. The Architecture Design level possesses four workflow details: Select candidate architecture, Define the abstract architecture, Refine the architecture, and Analyze behavior. The Detailed Design level also possesses four workflow details: Design components, Design connectors, Select components, and Select connectors.

The workflow details of the proposed model were tested, experimentally, in the SIG@UFPE project (developed by NTI/UFPE) to illustrate their use, detect inconsistencies, and verify the coherence of the activities and steps defined in each workflow detail.

ArcADe didn't encompass the whole software development process, because it focuses on the analysis and design stage. However, other stages of the process can also be affected and, consequently, need to be adapted. For example, the implementation stage must be redefined, because if a preexisting architectural element is selected, there should be adaptation (and not an implementation) effort.

Only one scenario was used to illustrate the execution of the proposed model, limiting the verification of ***ArcADe*** to just six workflow details. Consequently, the creation of new scenarios to exercise and verify the other workflow details, as well as the definition of the other stages of the software development process, are research work to be analyzed in future, so that we can have a complete development process based on software architecture.

References

[1] Alves, C.: Seleção de Produtos de Software Utilizando uma Abordagem Baseada em Engenharia de Requisitos. Dissertação de Mestrado. Universidade Federal de Pernambuco – UFPE. Mar, 2001.

[2] Bass, L., Clements, P., Kazman, R.: *Software Architecture in Practice*. Addison-Wesley, 1998.

[3] Catalysis. Website: http://www.catalysis.org, All contents copyrighted © 1998. Last access in Fev/2004.

[4] Dashofy, E., Medvidovic, N., Taylor, R.: Using Off-The-Shelf Middleware to Implement Connectors in Distributed Software Architectures. 1998.

[5] Egyed, A., Grunbacher, P., Medvidovic, N.: Refinement and Evolution Issues in Bridging Requirements and Architecture – The CBSP Approach. In *Proc. of the From Software Requirements to Architectures Workshop (STRAW 2001)*, Toronto, Canada, May 14, 2001.

[6] Gamma, E., Helm, R., Johnson, R., Vlissides, J.: *Padrões de Projeto – Soluções Reutilizáveis de Software Orientado a Objetos.* Porto Alegre: Bookman, 2000.

[7] Garlan, D., Kompanek, A., Cheng, S.: Reconciling the Needs of Architectural Description with Object-Modeling Notations. http://www-2.cs.cmu.edu/~able/publications/uml01/. Last access in Fev/2004.

[8] Jacobson, I., Griss, M., Jonsson, P.: *Software Reuse – Architecture, Process and Organization for Business Success*. Addison-Wesley, 1997.

[9] Krutchen, P.: *The Unified Software Development Process, An Introduction*. Addison Wesley, 1999.

[10] MDA Website: http://www.omg.org/mda, Last Updated Tuesday, March 12, 2002. Last access in Fev/2004.

[11] Medvidovic, N., Taylor, R.: A Classification and Comparison Framework for Architecture Description Languages. *IEEE Transactions on Software Engineering*, Vol 26. Nº 1, Jan, 2000.

[12] Moraes, M.: Um Framework de Análise e Projeto Baseado em Arquitetura de Software. Dissertação de Mestrado. Universidade Federal de Pernambuco – UFPE. Mar, Jul, 2002.

[13] Moriconi, M., Qian, X., Riemenschneider, R.: Correct Architecture Refinement. *IEEE Transactions on Software Engineering*, April, 1995, Vol.21, Nº 4, pp. 356-372.

[14] Rosa, N., Justo, G., Cunha, P.: A Framework for Building Non-Functional Software Architectures. In *16th ACM Symposium on Applied Computing*, Las Vegas, USA, 2001. 16th ACM SAC. , 2001. p.141 – 147

[15] Shaw, M., Garlan, D.: *Software Architecture: Perspectives on an Emerging Discipline*. Prentice Hall, New Jersey, 1996.

[16] Siegel, J.: *CORBA Fundamental end Programming*. John Wiley & Sons, Inc.1996.

[17] Spitznagel, B., Garlan, D.: A Compositional Approach for Constructing Connectors. In *Proc. of the Working IEEE/IFIP Conference on Software Architecture* (WICSA'01), 2001.

Una Propuesta de Integración de Animación Facial y Voz Sintética

José F. Ferreira
Universidad de Los Andes, Departamento de Sistemas y Computación,
Bogotá D.C., Colombia
j-ferrei@uniandes.edu.co

y

Fernando De la Rosa
Universidad de Los Andes, Departamento de Sistemas y Computación
Bogotá D.C., Colombia
fde@uniandes.edu.co

Abstract

The work presented in this document consists of an exploration of modern computer facial animation techniques. The goal of our research is the analysis, evaluation and study of feasibility for integration of these techniques with existing text to speech translation tools. This integration provides the user a method of virtual characters creation that allows the generation of facial movements which are synchronized with the synthetic speech generated automatically by the text to speech engine. As a base for the research, the use of the frame provided by the compression and transmission of multimedia standard MPEG-4 was decided. This standard includes a specification of the concepts applicable to computer facial animation. The results obtained from the evaluated techniques about the generated animation quality are satisfactory and demonstrate the possibility of use of the generated virtual characters with voice in computer applications as an user interaction metaphor.
Keywords: Facial Animation, Computer Animation, MPEG-4, Synthetic Voice, Text To Speech Engines.

Resumen

El trabajo presentado en este documento consiste en una exploración de las técnicas de animación facial actuales. El objetivo de nuestra investigación es el análisis, evaluación y estudio de factibilidad de la integración de estas técnicas con herramientas existentes de *traducción de texto escrito a voz sintética*. Esta integración provee al usuario de un método de generación de personajes virtuales que permite la generación de movimientos faciales sincronizados con la voz sintética generada automáticamente por el motor de traducción de texto a voz sintética. Para el desarrollo de la investigación se eligió la utilización del marco provisto por el estándar de compresión y transmisión de multimedia MPEG-4 que incluye una especificación de los conceptos aplicables a la animación facial. Los resultados obtenidos de la evaluación de las técnicas estudiadas sobre la calidad de la animación generada son satisfactorios y demuestran la posibilidad del uso de personajes virtuales animados, con voz, en aplicaciones computacionales como metáfora de interacción con el usuario.
Palabras claves: Animación Facial, Animación por computador, MPEG-4, Voz Sintética, Motores de traducción de Texto a Voz.

1. Introducción

Uno de los factores más importantes en el crecimiento de la computación en las últimas dos décadas ha sido la introducción y desarrollo de técnicas de digitalización de medios audiovisuales. Con el aprovechamiento de las capacidades multimedia de los computadores actuales se ha logrado transmitir, con efectividad y de manera dinámica, diferentes tipos de información de manera simultánea captando una mayor atención del usuario hacia el mensaje.
Sin duda alguna, la posibilidad de usar sonidos, imágenes, videos, animaciones y otras formas de comunicación en su forma digital, le ha dado un gran impulso a la industria informática de nuestro tiempo. Particularmente, en el área de la informática gráfica muchos de los esfuerzos realizados han sido destinados a lograr una representación realista del mundo real a través del uso de información digital.
Uno de los elementos fundamentales de la representación del mundo real en un ambiente digital es la representación de nosotros mismos; por esta razón, una de las áreas de investigación más activa en el campo de la computación gráfica es la animación facial de modelos geométricos para la representación de rostros.
Las técnicas desarrolladas por estas investigaciones han logrado ser aplicadas con éxito en diversas áreas tales como entretenimiento y educación. Hoy en día, este tipo de avances, junto al mejoramiento sustancial del hardware gráfico disponible en el mercado, ha permitido el reciente incremento de la utilización de personajes virtuales realistas en aplicaciones de diferente tipo como películas cinematográficas, software educativo, publicidad, entretenimiento, ambientes virtuales (chat, centros comerciales,…), etc.
La investigación presentada en este trabajo explora los conceptos en los que se basan las técnicas de animación facial actuales (representación geométrica, interpolación, reproducción de fonemas,…). A partir del conocimiento de éstos, se evalúa la factibilidad de la integración de herramientas existentes de traducción de texto escrito a voz sintética con el fin de proveer al usuario de un método de generación de personajes virtuales que, además de tener una voz propia, tengan un movimiento facial sincronizado con los sonidos generados automáticamente.
El desarrollo de cualquier aplicación resulta mucho más útil cuando es compatible con alguno de los estándares definidos en el mercado. Por este motivo, nuestra decisión fue el realizar un estudio de las características definidas en el estándar MPEG-4 para el soporte de la animación facial.
El presente artículo está organizado de la siguiente manera: la sección 2 presenta los aspectos del estándar MPEG-4 para animación facial. En la sección 3 se exponen la problemática de deformación geométrica facial y su sincronización con la voz sintética. A continuación, se describe la arquitectura de la solución propuesta y una aplicación computacional de la misma. Finalmente se encuentran los trabajos futuros y las conclusiones.

2. El Estándar MPEG-4 para Animación Facial

MPEG-4 define en detalle los parámetros para la definición y animación de modelos faciales [11, 12]. Por un lado, los parámetros para la definición facial (*FDP: Facial Definition Parameters*) permiten una especificación completa de la forma, tamaño y textura del modelo de la cara. Por otro lado, los parámetros de animación facial (*FAP: Facial Animation Parameters*) hacen posible la representación de expresiones faciales y visemas (i.e. conjunto de apariencias visuales relacionadas con un fonema; también llamados "*fonemas visuales*") mediante la manipulación de puntos característicos del modelo geométrico. Los *FAPs* son expresados en términos de *FAPUs* (*Facial Animation Parameter Units*) que consisten en la representación de la distancia entre los diferentes puntos característicos definidos para conservar una proporcionalidad adecuada al tamaño del modelo geométrico.
En el caso de la definición de los modelos faciales, los *FDPs* son usados para personalizar un modelo genérico (Ver Figura 1). Los campos definidos para una definición completa del modelo son las coordenadas de los puntos característicos, las coordenadas de la textura aplicada al modelo y el comportamiento de los parámetros de animación dentro de la escena [9, 13, 23].
El estándar MPEG-4 utiliza métodos de animación basados en parametrizaciones de la posición de un conjunto de puntos característicos predefinidos. En la especificación del estándar MPEG-4 se definen los ya mencionados *FAPs* que consisten, en la mayoría de los casos, de un desplazamiento unidireccional o bidireccional del punto característico asociado.
La definición de estos parámetros se basa en el estudio de las acciones musculares del rostro humano y comprende 68 parámetros divididos en dos grupos: parámetros de animación de bajo y de alto nivel [2, 11, 13]. Los *FAPs* de bajo nivel están relacionados con los puntos característicos definidos por los *FDPs* y generalmente son una medida de desplazamiento relativo de estos puntos dentro del modelo. Los *FAPs* de alto nivel comprenden las expresiones y los visemas. Estos dos últimos *FAPs* pueden mover al mismo tiempo un conjunto de puntos característicos modificando por completo el gesto del rostro. En particular, los visemas manipulan la postura y la posición de los labios para la

representación visual de la vocalización de un fonema [2, 5]. En el caso de las expresiones, se manipulan varios subconjuntos de *FAPs* de bajo nivel para obtener la representación de una expresión facial.
Debido a que estos parámetros deben poder ser usados en modelos geométricos de diferentes tamaños y características se definen también las *FAPUs*. Estas unidades son definidas como fracciones de distancias con respecto a características claves del rostro en un estado neutral. Los valores de los *FAPs* están dados en términos de *FAPUs* para lograr un desplazamiento adecuado en modelos de diferentes tamaños [12,13].

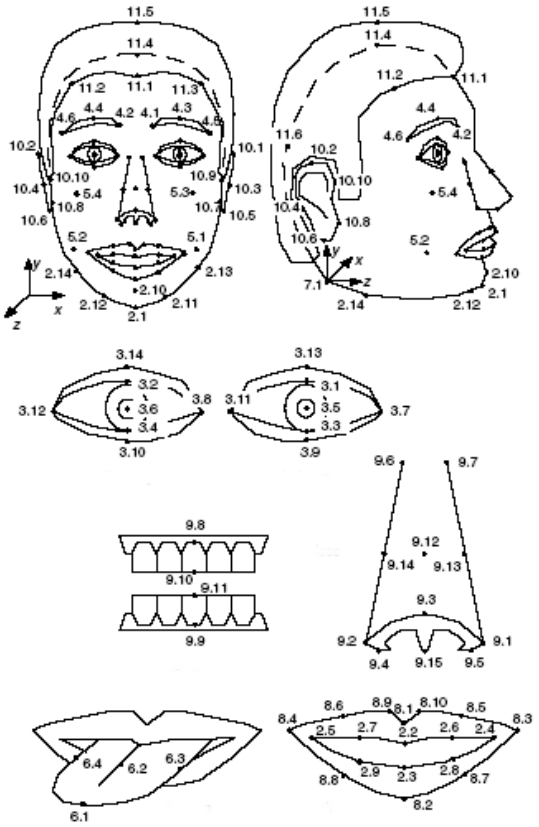


Figura No. 1. MPEG-4 ha definido un conjunto de 84 puntos característicos usados para calibrar y animar un rostro. Los puntos son clasificados de acuerdo a la región del rostro a la que pertenecen. [17]

Para producir un rostro animado compatible, MPEG-4 no especifica ninguna condición sobre el modelo utilizado a excepción que los puntos característicos definidos para el rostro sean desplazados de acuerdo a las magnitudes definidas por los *FAPUs* particulares del modelo geométrico durante la interpretación de los *FAPs* de bajo nivel.

3. Principales problemáticas identificadas

El principal problema que resuelve nuestra investigación es el análisis, diseño, implementación y pruebas de un sistema computacional capaz de crear un personaje virtual realista (representado por un modelo geométrico facial 3D) con la capacidad de realizar movimientos faciales sincronizados con una voz sintética generada a partir de un texto escrito.
De manera adicional, el sistema debe permitir que el personaje virtual creado sea una representación realista de un rostro de una persona real de manera que sea más natural la utilización de la aplicación para el usuario final.
A continuación se explica cada uno de los problemas que requieren ser tratados para lograr una solución completa, indicando las soluciones específicas propuestas para cada uno de ellos en la aplicación de prueba.

3.1 Deformación del modelo facial usando parámetros MPEG-4

La deformación del modelo facial es quizás el problema de mayor importancia en el alcance de esta investigación. El modelo facial debe ser deformado para generar las expresiones faciales declaradas por los parámetros de animación. En la solución de este subproblema se debieron identificar aquellas técnicas de animación facial que permiten el desarrollo

de una aplicación compatible con MPEG-4. Con el propósito de determinar qué enfoque existente de los usados en otras investigaciones resulta más adecuado para lograr el grado de compatibilidad deseado fue necesario relacionar los conceptos investigados sobre la animación facial con los principios establecidos por MPEG-4 para el manejo de este tipo de animaciones. Para lograr una deformación facial adecuada y lo más realista posible, es necesario estudiar y resolver los siguientes subproblemas específicos:

- La representación del modelo geométrico: Para lograr que un método de representación geométrica de la cabeza humana sea compatible con MPEG-4 basta con que el modelo tenga un vértice asociado con cada punto característico definido en MPEG-4. No es necesario incluir todos los puntos estandarizados dentro de una aplicación para lograr compatibilidad con MPEG-4 [12, 23]. Adicionalmente, para cada punto característico se deben definir sus respectivas coordenadas de textura.

- La selección interactiva de los puntos a deformar: Con el objetivo de solucionar la problemática de selección de puntos característicos, MPEG-4 introduce el concepto de *FDPs*. Estos parámetros pueden ser usados ya sea para modificar la forma y apariencia de un modelo o para codificar la información necesaria en la transmisión de un modelo completo junto con los criterios que deben ser usados para animarlo. Los *FDPs* son usados con el fin de cambiar la ubicación relativa de los puntos característicos definidos por el estándar MPEG-4 [12]. Sin embargo, dicho estándar no define un método específico de generación y codificación de *FAPs* ni *FDPs* [1], por lo que cada aplicación puede definir un método interno de representación de estos parámetros siempre y cuando la transmisión de esta información hacia el exterior de la aplicación siga las normas semánticas y sintácticas acordes a MPEG-4 [12]. Para el caso específico de nuestra aplicación de prueba, el soporte de este tipo de transmisión externa no fue necesario, por lo que el único aspecto a resolver consiste en la definición de la representación interna de la información necesaria para la animación del modelo facial en la aplicación.

- La deformación de grupos de vértices: Cuando el modelo es modificado durante una animación es necesario que los vértices cercanos a aquel que representa un punto característico, sean también movidos consistentemente con el fin de lograr un mayor realismo [13].

- La interpolación de los parámetros de animación usados: Los *FAPs* que controlan la animación son valores fijos definidos para un instante de tiempo preciso, y por lo tanto una animación que considere únicamente los movimientos para estos instantes de tiempo se vería poco realista (i.e. se apreciarían cambios bruscos entre los instantes definidos). Por este motivo es necesaria la inclusión de técnicas de interpolación de animación para los instantes de tiempo que se encuentran entre dos cuadros de animación definidos por dos *FAPs* consecutivos.

En resumen, para ofrecer una solución completa al problema de animación facial, es necesaria la inclusión de técnicas adecuadas de selección de puntos en un modelo facial 3D que permitan al usuario escoger los *FDPs* de una representación geométrica 3D compatible con MPEG-4, así como una forma de determinar el conjunto de vértices que se ven afectados debido a la deformación de un punto característico y que es definida por los *FAPs*. También es necesario definir los métodos de interpolación de animación de cuadros para suavizar los movimientos determinados por los *FAPs* que son parámetros discretos de animación, razón por la cual no definen el estado del modelo en todos los instantes posibles de tiempo.

3.2 Sincronización de voz sintética y animación

Una característica fundamental de nuestra investigación es la sincronización de la animación con la voz sintética generada por los motores de traducción a partir de un texto escrito. Dicha voz sintética debe corresponder con los movimientos realizados por los parámetros de animación facial para preservar un nivel de realismo aceptable. Debido a que los módulos de animación y síntesis de voz son independientes, el control del tiempo de ejecución debe ser resuelto en la etapa previa a la generación de las ondas de audio a partir del texto de entrada.

Los sistemas de síntesis de voz generada a partir de un texto deben proveer al sistema de animación facial tanto de la información sobre los fonemas que deben ser representados como su aparición en el tiempo en la onda de audio previamente generada. En el caso de la especificación definida por el estándar MPEG-4, aunque se contempla la integración de motores TTS (*Text To Speech*) con el resto de las tecnologías a través de una interfaz genérica llamada TTSI (*Text To Speech Interface*) [4], no se aclara la utilización de información de sincronización entre el audio y la información de animación. Por este motivo, es necesario definir una arquitectura externa que genere un flujo sincronizado de las dos fuentes, el motor TTS y la interfaz de animación del modelo. Ostermann *et al.* [14] proponen una arquitectura de sincronización basada en *FAPs* usando un mecanismo de marcas de tiempo (llamadas *"bookmarks"*) y una función de interpolación de puntos para determinar el estado del modelo entre las marcas de tiempo. Para la sincronización de las salidas de audio y animación, se utiliza la información de los tiempos de reproducción de los fonemas generados por el motor de traducción para calcular los cuadros de la animación del modelo facial.

La propuesta de Ostermann *et al.* [14] se usó como base para la definición de la arquitectura de la aplicación de prueba desarrollada durante nuestra investigación en cuanto a las ideas expuestas sobre el proceso de transformación de la información proveniente del motor de traducción TTS en parámetros MPEG-4 utilizables por el módulo encargado de la generación de la animación.

4. Implementación de la solución en la Aplicación de Prueba

Con el fin de validar la aplicabilidad de las técnicas estudiadas en un caso real y de evaluar las soluciones encontradas durante la investigación a las problemáticas definidas en la sección anterior se implementó una aplicación de prueba en lenguaje C/C++ usando el API de programación OpenGL [22]. A continuación se exponen algunos detalles de la estructura del sistema desarrollado.

4.1 Arquitectura general del sistema

En la Figura No. 2 se muestra el esquema general de la arquitectura del sistema propuesto. En el esquema se pueden identificar claramente tres módulos:

- Módulo de Definición Facial encargado de la definición del modelo geométrico a través del soporte del formato OBJ y la representación de los puntos característicos compatibles con MPEG-4 que serán definidos de manera interactiva por el usuario (Figura No. 3).
- Módulo de Animación Facial, que se ocupa del manejo de los parámetros de animación y su correspondiente efecto en el estado de los vértices del modelo incluyendo los aspectos de interpolación entre los cuadros de animación definidos por los parámetros y de deformación de vértices vecinos a un punto característico movido (Figura No. 4).
- Módulo de Decodificación de Texto y Síntesis de Voz, que tiene como función principal controlar la generación de los parámetros de animación de acuerdo a la información proveniente de la decodificación del texto escrito (Figura No. 5). El proceso de decodificación y síntesis de voz se realizará con el Festival Speech Synthesis System [20].

Al interior de la aplicación (Figura No. 2), el componente de sincronización recibe por separado los parámetros de animación y los de voz sintética y reproduce las señales video y audio de manera simultánea.

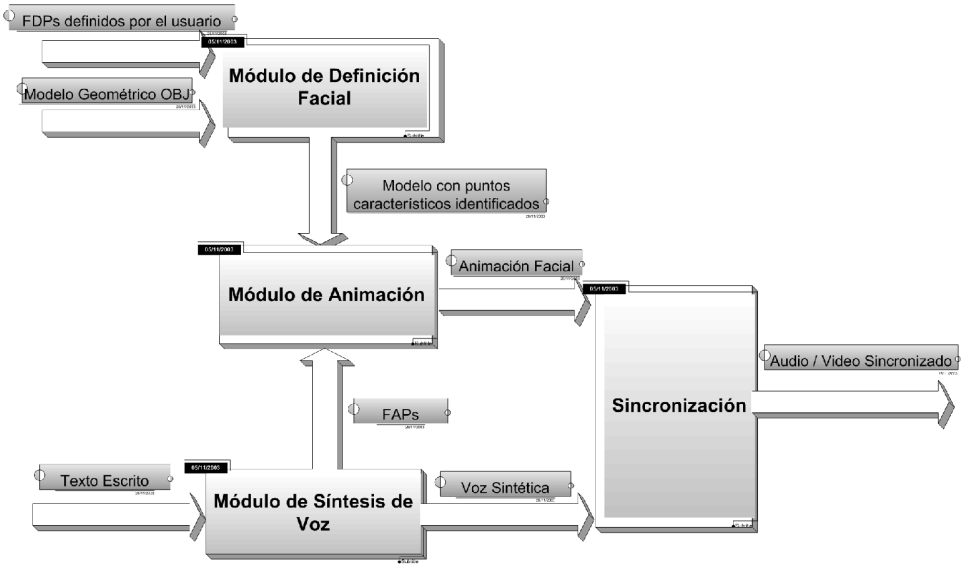


Figura No. 2. Esquema general de la arquitectura del sistema propuesto

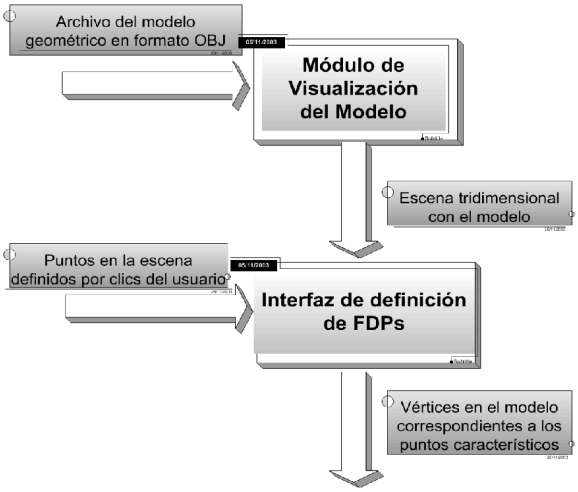


Figura No. 3. Módulo de definición del modelo facial.

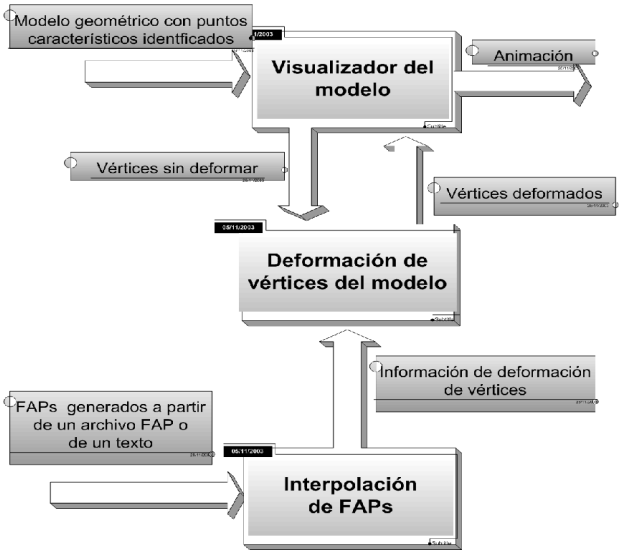


Figura No. 4. Módulo de animación facial.

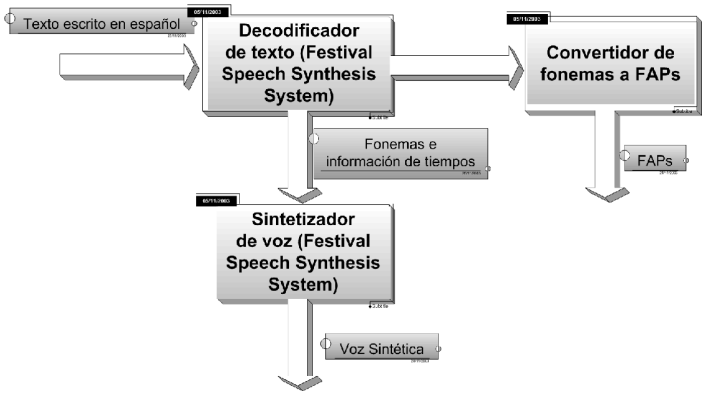


Figura No. 5. Módulo de decodificación de texto y síntesis de voz.

4.2 Despliegue del modelo y selección de puntos característicos

Para la inclusión de la selección de puntos característicos en la aplicación de prueba se utilizó el mecanismo de *color coding* [16]. Esta alternativa se eligió debido a la facilidad de implementación y a la independencia que ofrece de la posición del observador y del estado del modelo que en los otros métodos evaluados añaden cierto grado de complejidad y dificultan una detección adecuada del vértice seleccionado.

El mecanismo de *color coding* se basa en la utilización de un código de colores diferentes para cada elemento seleccionable del modelo. En el caso específico de la aplicación de prueba se utiliza una codificación especial sobre los valores RGB del color de los vértices de manera que cuando el usuario hace *clic* con el ratón sobre uno de los vértices del modelo, se obtienen los valores RGB del píxel y se transforman en el identificador del vértice en el modelo de representación interna.

4.3 Deformación del modelo geométrico

El control de la animación generada es establecido por los parámetros *FAP* como lo establece el estándar MPEG-4. En la aplicación de prueba se definen las estructuras que guardan la información proveniente de archivos *FAP* o del análisis de la información de síntesis de voz. Posteriormente, esta información es usada para determinar los puntos característicos a deformar en el modelo y lograr la animación del modelo facial.

Se decidió aplicar el mecanismo más sencillo posible de deformación de vértices vecinos y evaluar el desempeño y nivel de realismo de la animación lograda. Por esta razón, la aplicación utiliza un método de deformación constante de grupos predefinidos de vértices para lograr la animación del modelo.

El proceso de deformación implementado consta, básicamente, de dos etapas:

- Inicialización: Se definen los grupos de vértices a partir de los puntos característicos definidos por el usuario sobre el modelo. No todos los puntos característicos determinan la creación de un grupo de vértices, sólo aquellos que se consideran esenciales para una correcta apariencia de los movimientos básicos del rostro. El detalle de los grupos de labios a partir de subgrupos se determinó por el efecto que éstos tienen en la apariencia realista del movimiento involucrado en la gesticulación de visemas. Adicionalmente se calculan las unidades *FAPU* correspondientes al modelo a partir de los puntos característicos definidos por el estándar MPEG-4.

- Deformación en tiempo real: El conjunto de vértices asociado a un punto característico seleccionado se deforma en una proporción igual al movimiento (i.e. cambio) de este punto característico. Este método de deformación, contrario a lo que se podría pensar, logra un realismo de movimiento aceptable (en deformaciones con magnitudes relativamente pequeñas) unido a un desempeño superior comparado con el de otras alternativas evaluadas (métodos de *warping* [15,18], *free form deformations* [6, 7, 10, 21], deformaciones basadas en los puntos característicos [8]) que involucran cálculos más intensivos en términos computacionales.

4.4 Interpolación de parámetros de animación

La definición de cada cuadro de animación mediante los parámetros de animación no es suficiente para la obtención de una animación realista debido a que entre dos cuadros consecutivos de animación no se tiene información sobre la posición de los vértices del modelo. La solución a este problema consiste en la inclusión de técnicas de interpolación temporal o *in-betweening* de los cuadros de animación. Cuando la aplicación intenta mostrar el estado del modelo facial en un tiempo situado entre dos cuadros de animación se calcula un valor intermedio, entre los parámetros del cuadro inicial y los parámetros del cuadro consecutivo, que depende del tiempo transcurrido en la animación.

Después de evaluar las diferentes posibilidades para la implementación de interpolación temporal entre dos cuadros de animación se decidió utilizar una función lineal definida para el cambio de posición de los vértices en el tiempo. Este cálculo se realiza para cada ciclo de visualización *(i.e. rendering)* de la escena, usando como parámetro de la función de interpolación el tiempo transcurrido desde el inicio de la reproducción de la animación.
La elección de esta alternativa se basó fundamentalmente en la intención de liberar a la aplicación de prueba de cálculos más complejos obteniendo de todas formas una buena solución. El análisis de funciones de interpolación más complejas determinó que éstas no introducían una mejora significativa en la calidad de la animación generada.

4.5 Sincronización de las salidas del sistema
Desde el inicio de la investigación se determinó que para la inclusión de un módulo de decodificación y síntesis de voz se debería utilizar software ya existente en el mercado debido a la alta complejidad de las posibles soluciones a esta problemática.
Por ésta razón uno de los estudios realizados durante nuestra investigación consistió en la evaluación de los diferentes sistemas de decodificación y síntesis disponibles para determinar cuál podría ser utilizado.
Como resultado se encontró que trabajos relacionados con el que se presenta en este documento [3,19] han utilizado con éxito las funcionalidades ofrecidas por el Festival Speech Synthesis System [20] para la traducción de texto en fonemas con la ventaja sobre otras soluciones de que esta aplicación está disponible de manera gratuita para fines investigativos.
El Festival Speech Synthesis System soporta el español y provee un API en C/C++ que tiene la capacidad de recibir como parámetros de entrada un texto y generar una trascripción equivalente de los fonemas contenidos en él junto con información de la duración de cada uno de ellos. Esta información es analizada por un lado para producir la onda de audio que será reproducida por la aplicación e igualmente para la traducción de fonemas y marcas de tiempo a *FAPs* lo que asegura la sincronización del audio y la animación.
La traducción de fonemas a *FAPs* se basa en el siguiente método. Debido a que la animación tiene una velocidad predefinida dada en cuadros por segundo y la información de inicio de los fonemas y su duración está disponible, se realiza una traducción de los fonemas a parámetros *FAP* de alto nivel, codificando el visema correspondiente en el cuadro de la animación más cercano al tiempo especificado por la información de síntesis. Debido a que esta aproximación introduce un pequeño desfase en la sincronización de la animación y la voz, se realiza un proceso de interpolación adicional entre dos cuadros que contienen visemas para generar la información intermedia en términos de *FAPs* de bajo nivel.
Otro de los puntos a resolver para la solución de esta problemática al interior de la aplicación de prueba, fue el de lograr una correspondencia adecuada entre los visemas definidos por el estándar MPEG-4 y los fonemas resultado del análisis del texto de entrada en Español realizado por el motor de traducción de texto de Festival. La propuesta de traducción especificada se puede ver en la Tabla No. 1.

	VISEMAS MPEG-4	***FONEMAS FESTIVAL ESPAÑOL***
0	Ninguno	#
1	p, b, m	P, b, m
2	f, v	F
3	T, D	Ninguno
4	t, d	t, d
5	k, g	k, g
6	tS, dZ, S	x, ch, ll
7	s, z	s, th (z)
8	n, l	n, ny (ñ), l
9	R	r, rr
10	A	a, a1
11	E	e, e1
12	I	i, i0, i1
13	Q	o, o1
14	U	u, u0, u1

Tabla No. 1. Correspondencia entre los visemas definidos por MPEG-4 y los fonemas generados por Festival a partir del texto de entrada en lenguaje Español.

Con la integración de las soluciones propuestas a las problemáticas definidas se consigue obtener una aplicación con las funcionalidades esperadas. El esquema general de funcionamiento de la aplicación y el proceso de transformación de la información es mostrado en la Figura No. 6. Adicionalmente, la Figura No. 7 muestra la aplicación implementada basándose en las técnicas expuestas en este artículo.

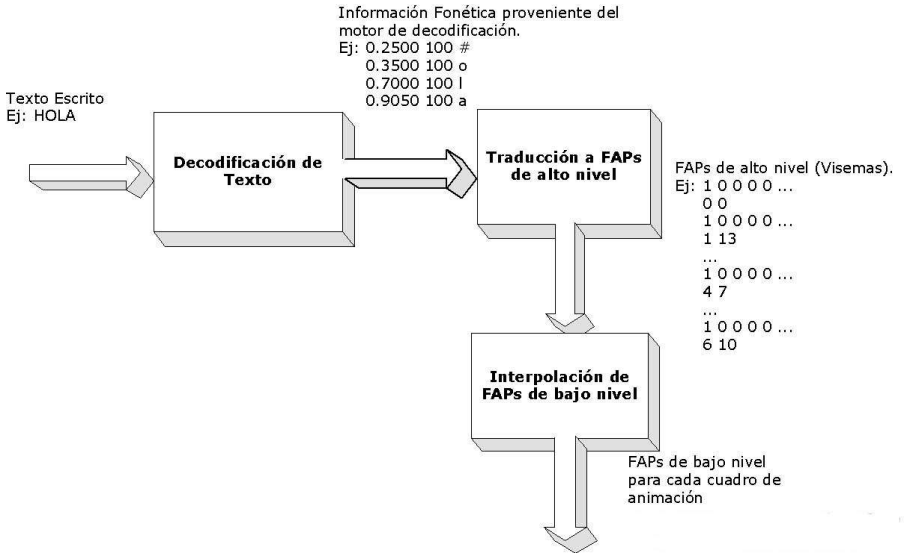


Figura No. 6. Proceso de sincronización de las salidas de audio y animación

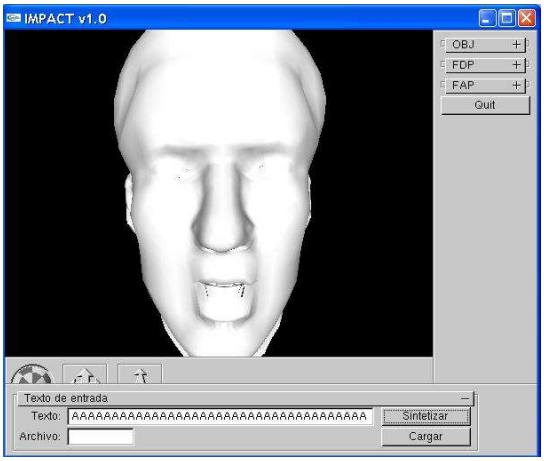


Figura No.7. Interfase de la Aplicación de Prueba

5. Trabajo futuro

Nuestra investigación cumplió con el objetivo de proponer una solución al problema de integrar la Animación Facial con Voz Sintética. Sin embargo, existe una gran variedad de estudios que podrían desarrollarse para dar un tratamiento más amplio al tema:

5.1 Posibles aplicaciones de los conceptos desarrollados

Uno de los temas de investigación que probablemente tendría mayor importancia sería la definición de aplicaciones que utilicen las funcionalidades definidas en la aplicación de prueba para fines específicos. Entre los tipos de aplicaciones que se podrían desarrollar se encuentran:

- Ambientes Virtuales Compartidos (accesibles a través de Internet) como por ejemplo una aplicación de pseudo-teleconferencia o "chat visual" que a través de la utilización de personajes virtuales facilite el diálogo entre dos o

más personas al usar información tanto visual como sonora transmitida en términos de parámetros de animación y procesada de manera local para la generación de la animación.

- Interfaces antropomórficas que agreguen un componente adicional de facilidad de uso y de interacción con el usuario mediante la representación de un personaje virtual que sea el encargado de comunicar los diferentes eventos al usuario. Este tipo de interfaces sería muy útil en aplicaciones educativas o informativas en los que los usuarios carezcan de un conocimiento extenso de la manipulación del sistema o sean personas con discapacidades físicas que no les permitan una interacción tradicional con el sistema.
- Aplicaciones de generación de animaciones para uso en juegos u otros medios audiovisuales que faciliten la introducción de personajes virtuales resolviendo automáticamente la problemática de sincronización de labios con las voces *(lip sync)*.
- Aplicaciones variadas de entretenimiento en las que una de las formas de interacción con el sistema sea la introducción de texto. Por ejemplo, una aplicación que soporte la visualización de una historia en la que el usuario sea partícipe y modifique o determine los sucesos que ocurren durante el desarrollo de la trama.

5.2 Extensiones a las funcionalidades de la aplicación de prueba

El trabajo realizado durante la investigación deja abiertas algunas alternativas para un desarrollo más complejo de algunas de las características de la aplicación de prueba:

- Implementación de un módulo propio de obtención de los modelos faciales a partir de fotografías que permitiría un mejor control sobre las propiedades del modelo mediante la introducción de un modelo genérico con puntos característicos predefinidos que sirva como base para la creación de los modelos personalizados. También sería interesante estudiar la factibilidad de desarrollo de módulos de obtención de modelos a partir de otras fuentes como el vídeo y la detección automática de los puntos característicos del rostro.
- Implementación de animación corporal (también especificada en el estándar MPEG-4) de manera análoga a la animación facial presentada.
- Integración con otros sistemas que amplíen la gama de usos de la aplicación. Dentro de este tipo de sistemas se encuentran los sistemas de reconocimiento de voz que integrados con los resultados actuales permitiría generar directamente una animación facial a partir de la voz de una persona. Igualmente la integración con sistemas de inteligencia artificial basados en agentes (i.e. entidades autónomas) capaces de mantener una conversación con un usuario mediante la generación de textos de entrada.
- Soporte de formatos de definición de modelos geométricos variados (actualmente hay soporte de modelos OBJ).
- Generación de salidas en formatos de animación o vídeo usados de manera amplia, especialmente los definidos por el estándar MPEG-4. Dichas salidas podrían ser visualizadas en otras aplicaciones que soporten dichos formatos.
- Integración de otras técnicas de deformación para mejorar el nivel actual de realismo de las animaciones generadas.

6. Conclusiones

La primera etapa de nuestra investigación consistió en la comprensión de los conceptos en los que se fundamenta los temas de animación facial y síntesis de voz. Posteriormente, investigamos y analizamos los métodos necesarios para desarrollar un sistema computacional que resuelven e integran las dos problemáticas. La implementación de los métodos finalmente escogidos se integraron en la aplicación de prueba, resultado de este trabajo.

Las evaluaciones practicadas hasta el momento cumplen los objetivos inicialmente previstos. Sin embargo, el nivel de los resultados puede mejorar mediante su aplicación en áreas específicas (ambientes virtuales compartidos, entretenimiento, educación, publicidad,...).

Uno de los factores más importantes dentro del desarrollo de la investigación fue la elección del estándar MPEG-4 ya que éste provee la definición precisa de un marco que resulta apropiado para la construcción de aplicaciones de animación computacional y, de cierta manera, establece un esquema general que resulta muy útil para la definición de una arquitectura apropiada. Adicionalmente, MPEG-4 suministra una forma de controlar animaciones relativamente complejas (como lo son las animaciones faciales) a partir de un flujo simple de datos. Dicho flujo permite una definición simple de interacción entre los diferentes módulos de la aplicación y una integración sencilla con el software de síntesis de voz que, en principio, es uno de los objetivos primordiales de este proyecto.

Otro de los factores relevantes en la problemática estudiada es la elección del motor de decodificación y síntesis de voz. En nuestro trabajo, Festival Speech Synthesis System resultó ser una buena elección debido a la variedad de opciones de control de las salidas de información fonética y voz que genera. Esto hace posible el proceso de sincronización animación – voz sintética. Vale la pena mencionar que tanto Festival como otros motores del mismo tipo no generan (por ahora) voz humana natural.

Referencias

[1] Antunes-Abrantes, G. y Pereira, F. MPEG-4 Facial Animation Technology: Survey, Implementation and Results, *IEEE Transactions on Circuits and Systems*. Vol. 9, No. 2, (Marzo 1999), pp. 290-305.

[2] Eptamedia, Facial Animation Background, *http://www.eptamedia.com/en-doc/en-faceanim-c-background.htm*. (Revisado en Noviembre 2003)

[3] Huynh, H.Q. A Facial Animation Markup Language (FAML) for the Scripting of a Talking Head. Curtin University of Technology, (2000).

[4] ISO/IEC JTC1/WG11 N2201, Texto de la especificación ISO/IEC FCD 14496-1: Systems. 1998.

[5] Joslin, C., Molet, T., Magnenat-Thalmann, N. Distributed Virtual Reality Systems. MIRALab, CUI, University of Geneva, 2001.

[6] Kalra , P., Mangili, A., Magnenat-Thalmann, N., Thalmann D. 3D Interactive Free Form Deformations for Facial Expressions. MIRALab, University of Geneva, 1991.

[7] Kalra, P., Mangili, A., Magnenat-Thalmann, N., Thalmann, D. Simulation of Facial Muscle Actions Based on Rational Free Form Deformations. *Computer Graphics Forum*, Vol. 11, No. 3, (1992), pp. 59-69

[8] Kshirsagar, S., Garchery, S., Magnenat-Thalmann, N. Feature Point Based Mesh Deformation Applied to MPEG-4 Facial Animation. *Proceedings Deform'2000*; (2000).

[9] Lavaggetto, F. & Pockaj, R. The Facial Animation Engine: Towards a High-Level Interface for the Design of MPEG-4 Compliant Animated Faces. *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 9, No. 2, (March 1999), pp.277-289

[10] Martin, Suzanne. Free – form Deformation. *http://www.cs.wpi.edu/~matt/courses/cs563/talks/smartin/ffdeform.html* (Revisado en Agosto 2003)

[11] Moving Picture Experts Group, ISO/IEC 14496 – MPEG-4 International Standard, *www.cselt.it/mpeg/*. (Revisado en Agosto 2003)

[12] MPEG Video & SNHC, Texto de la especificación ISO/IEC FDIS 14496-2: Visual.

[13] Ostermann, J. Face Animation in MPEG-4. *MPEG-4 Facial Animation (I.S. Pandzic and R. Forchheimer, Eds.)*, p.p.17-56, Chichester, U.K., John Wiley \& Sons, (2002).

[14] Ostermann, J., Beutnagel, M., Fischer, A., and Wang, Y. Integration of Talking Heads and Text to Speech Synthesizers for Visual TTS. *International Conference on Speech and Language Processing*. Sydney – Australia, (1998), p. 297-300

[15] Parent, R. A System for Generating Three-Dimensional Data for Computer Graphics ,Ph.D. Dissertation, Ohio State University, (1977).

[16] Picking Tutorial OpenGL. http://www.lighthouse3d.com/opengl/picking/.(Revisado en Agosto 2003)

[17] Pockaj, R., Baudino, M.,Corte, F., Ambrosini, L., Costa M., Bonamico C. The Facial Animation Engine Demo. *http://www-dsp.com.dist.unige.it/~pok/RESEARCH/MPEG/fae.htm*. (Revisado en Octubre 2003)

[18] Porcher, L. Animacao por Computador – Deformacao. Instituto de Informática UFGRS. *http://www.inf.ufrgs.br/~nedel* (Revisado en Septiembre 2003)

[19] Stallo, J. Simulating emotional speech for a talking head. Curtin University of Technology, (2000).

[20] The Centre for Speech Technology Research - University of Edinburgh. Festival Speech Synthesis System. *http://www.cstr.ed.ac.uk/projects/festival/.* (Revisado en Marzo 2004)

[21] Uribe, D. Uso de Humanoides para Comercio Electrónico. Universidad de Los Andes, (2002).

[22] Woo, M. OpenGL programming guide : the official guide to learning OpenGL, version 1.2. 3rd ed. Boston, Mass. : Addison Wesley , c1999.

[23] Won-Sook, L., Escher, M., Sannier G., Magnenat-Thalmann, N. MPEG-4 Compatible Faces from Orthogonal Photos. Proceedings CA99 (International Conference on Computer Animation), Geneva, Switzerland. May 26-29, 1999, pp.186-194.

PROPUESTA Y EVALUACIÓN DE UN MODELO DE RECONFIGURACIÓN DINÁMICA EN UN SUBSISTEMA DE ENTRADA/SALIDA REDUNDANTE PARA UN SISTEMA DE ARCHIVOS DISTRIBUIDO Y PARALELO.

JUAN PABLO GARCÍA OJEDA

Universidad Austral de Chile, Ingeniero Civil en Informática, jgarcia@inf.uach.cl, Casilla 567 - Valdivia.

RAIMUNDO VEGA VEGA

Universidad Austral de Chile, Doctor en Informática, rvega@uach.cl, Casilla 567 - Valdivia.

Resumen **- El presente trabajo estudia el problema del almacenamiento masivo de información centrándose en la disponibilidad de los datos. Para esto, se toma como base un simulador de un sistema de archivos distribuido y paralelo con tolerancia a fallos al cual se le añadió una nueva funcionalidad conocida como reconfiguración dinámica, es decir la característica que permite al sistema poder agregar más nodos de almacenamiento sin necesidad de detener la normal entrega de servicios. Por último se generan pruebas que permiten analizar los resultados y compararlos con otros estudios realizados anteriormente sobre el mismo sistema bajo condiciones que no incluyen reconfiguración.**

Abstract **- The current work studies the problem of the massive storage of information being centered in data availability . For this purpose, it takes as base a fault tolerance distributed and parallel system simulator which has been added a new functionality known as dynamic reconfiguration, it means the attribute that allows to the system to be able to add more storage nodes without need to stop normal services delivery. Finally, tests are generated that allow to analyze the results and to compare them with other studies carried out previously on the same system under conditions that don't include reconfiguration.**

Palabras clave **– sistemas distribuidos, paralelismo, tolerancia de fallos, sistemas de archivos, redundancia, reconfiguración.**

1. INTRODUCCIÓN

La informática se define como: "El conjunto de conocimientos científicos y técnicos que hacen posible el tratamiento automático de la información por medio de ordenadores" [1]. Con el pasar de los años, esta ciencia ha aprendido que el valor de la información obtenida depende en gran manera de la cantidad y calidad de los datos que la subyacen. Existen ocasiones en que la cantidad de datos es tan abrumadora que el acceso a estos no se puede obtener en un lapso razonable de tiempo por lo que la información que con ella se obtiene se considera obsoleta.

Con el transcurrir del tiempo, la velocidad de procesamiento superó considerablemente a la velocidad de entrada y salida a los datos proporcionada por los sistemas de archivos, lo que trajo consigo la llamada "Crisis de la E/S" [2], la cual se agrava aún mas en entornos paralelos donde el acceso concurrente a los archivos, es decir, dos o más procesos acceden a este, es muy frecuente [3,4] y tanto los sistema de archivos tradicionales como los distribuidos no están optimizados para este tipo de acceso [5]. De esta problemática aparecieron los Sistemas de Archivos Distribuidos y Paralelos, los cuales combinan soluciones tales como: paralelismo en el sistema de entrada y salida, interfaces paralelas y caché en el sistema de entrada y salida.

Estos últimos sistemas han sido hasta el momento la solución a la crisis de la E/S. Sin embargo aún queda por incorporar el tema de la disponibilidad de datos, cuestión que se torna crítica en los sistemas de archivos distribuidos y paralelos ya que, a medida que se incrementa el paralelismo y la cantidad de nodos de entrada/salida, también aumenta la probabilidad de fallo del sistema [6]. En vista del poco estudio que había en este campo, es que en [6] se propuso un modelo de redundancia de datos que además fue implementado en un simulador especialmente diseñado para el efecto [6][8].

A partir del estudio que ahí se realizó, este trabajo estudiará un método que permita incorporar en forma dinámica nuevos dispositivos sin detener la normal entrega de servicios del sistema de entrada/salida.

2. DESCRIPCIÓN DEL PROBLEMA

Un sistema de archivos distribuido y paralelo, es aquel en el cual los datos se encuentran repartidos en múltiples nodos dentro de una red de computadores y estos son accedidos de manera simultánea para mejorar el rendimiento. Con el objetivo de estudiar el desempeño de estos sistemas, se ha construido un sistema simulador de este tipo de arquitecturas que además permite medir el ancho de banda y los tiempos de lectura y escritura para cargas de tipo científica y transaccionales (OLPT). El sistema es configurable mediante el ingreso de distintos parámetros que permiten identificar el tipo de distribución que tienen los elementos de este, tales como: el número de nodos en la red, tamaño de las unidades de reparto, nivel de RAID, etc.

Este trabajo se centra en la formulación de un modelo que incorpore una nueva funcionalidad al sistema y este pueda así incorporar nuevos nodos a la red de manera que la información se redistribuya de forma balanceada y sin que el sistema este fuera de servicio.

3. MODELO DE UN SISTEMA DE ARCHIVOS DISTRIBUIDO Y PARALELO CON REDUNDANCIA DE DATOS.

Un sistema de archivos distribuido y paralelo con redundancia de datos, es aquel en el cual la información que se encuentra distribuida dentro de los nodos incorpora información redundante, que le proporciona al sistema características de tolerancia a fallos(fig.1)

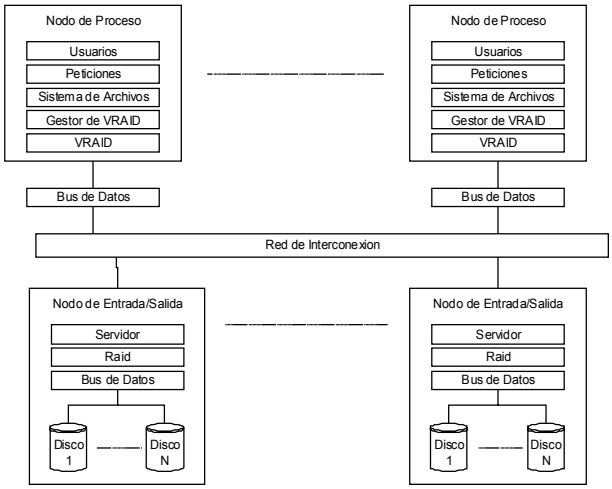


Figura 1. Esquema de un sistema de archivos distribuido y paralelo con redundancia de datos.

Es decir, el sistema es capaz de permitir el fallo de uno o más de los nodos dependiendo de la cantidad de información redundante que se desee incorporar sin perder la disponibilidad de la información. Una forma de alcanzar redundancia de datos en un sistema de archivo distribuido y paralelo es utilizando un esquema similar al empleado por las Matrices de Discos Redundantes, que se puede representar por una matriz de m filas y n columnas (Figura 2).

En este caso las n columnas representan el número de nodos de almacenamiento existentes dentro de la red y m el número de bloques de almacenamiento dentro de cada uno de los nodos. Cada fila de la matriz es conocida como Franja de Paridad y cada elemento de la matriz es un bloque dentro del sistema de archivos distribuido. En cada una de las franjas de paridad existe un bloque representado por una letra P que almacena la información redundante calculada en base al resto de los bloques que comparten la misma franja de paridad. De esta manera existen m*(n-1) bloques de almacenamiento.

	N0	N1	N2	N3
F0	0	1	2	P
F1	3	4	P	5
F2	6	P	7	8
F3	P	9	10	11
F4	12	13	14	P
F5	15	16	P	17
F6	18	P	19	20
F7	P	21	22	23
F8	24	25	26	P
F9	27	28	P	29

Figura 2. Matriz de un sistema de archivos distribuido y paralelo con tolerancia de fallos.

La arquitectura del modelo de sistema de archivos distribuido y paralelo con redundancia de datos con la cual se ha trabajado consta de las siguientes capas (fig3):

Figura 3. Bloques de un sistema de archivos distribuido y paralelo con redundancia de datos.

Trabajadores: Este módulo se encarga de simular la generación de todas las solicitudes de entrada/salida al sistema de archivos.

Sistema de Archivos Paralelo: Este módulo representa un sistema de archivos que lanza las peticiones hacia los dispositivos de las capas inferiores [7]. Para esto simula los procesos en que deben dividirse las solicitudes de entrada/salida con el fin de ser enviadas a los nodos respectivos.

Red: Modelo de una red de trabajo que interconecta nodos con CPUs, puede configurarse de diferentes formas, lo cual determina sus prestaciones. Interconecta y crea una cantidad de nodos que componen el sistema [7].

Servidores: Implementa un servidor de entrada/salida el cual esta encargado de monitorear los tiempos de transferencia por la red.

Raid: Modela un dispositivo Raid de niveles 0, 4 o 5, los que abarcan el conjunto de todos los discos de un nodo. Distribuye la operación solicitada en operaciones a los discos involucrados. Este módulo despacha las operaciones que se le solicitan en el orden en que llegan, de forma secuencial, pero explotando la concurrencia entre los discos [7].

Bus de Datos: Modelo de un bus de entrada/salida, puede configurarse de diversas formas que determinan sus prestaciones. Un bus transmite información desde un nodo a un disco o viceversa. Pueden existir varios buses por nodo [7].

Disco Duro: Modelo de un disco duro. Los discos pueden ser de distintos tipos lo cual determina sus prestaciones. Un disco ira conectado a un bus. Puede haber varios discos por bus [7].

Gestor de Virtual Raid: Un dispositivo gestionador de Raid que agrupa componentes de VRAID que permite al sistema redundante aumentar su disponibilidad de datos [7].

VRAID: Modela un dispositivo virtual Raid de niveles 0, 4 o 5. Este módulo despacha las operaciones que le solicitan los usuarios en el orden en que llegan, de forma secuencial, pero explotando la concurrencia entre los nodos[7].

3.1. Modelo del problema.

El problema se puede modelar como un arreglo de m filas y n columnas (Figura 4) en el cual las columnas representan los nodos del VRAID y las filas representan las franjas de paridad dentro de cada uno de nodos. Cada elemento de la matriz identifica un bloque con información, si el elemento esta representado por una P significa que ese bloque es utilizado para almacenar la información de paridad de la franja respectiva.

	*N*0	*N*1	*N*2	*N*3
*F*0	0	1	2	*P*
*F*1	3	4	*P*	5
*F*2	6	*P*	7	8
*F*3	*P*	9	10	11
*F*4	12	13	14	*P*
*F*5	15	16	*P*	17
*F*6	18	*P*	19	20
*F*7	*P*	21	22	23
*F*8	24	25	26	*P*
*F*9	27	28	*P*	29

Figura 4. Matriz de un sistema de archivos distribuido y paralelo con tolerancia de fallos.

El problema a resolver consiste en otorgarle al sistema la capacidad de incorporar nuevos nodos (Fig. 5), es decir agregar mas columnas a la matriz de manera que se reordenen los bloques dentro de la nueva matriz, cuidando que los bloques que almacenan información de paridad sean recalculados y no redistribuidos. Este proceso es el que se conoce como reconfiguración (Fig. 6).

	*N*0	*N*1	*N*2	*N*3	*N*4	*N*5
*F*0	0	1	2	*P*	–	–
*F*1	3	4	*P*	5	–	–
*F*2	6	*P*	7	8	–	–
*F*3	*P*	9	10	11	–	–
*F*4	12	13	14	*P*	–	–
*F*5	15	16	*P*	17	–	–

Figura 5. Matriz de un sistema de archivos distribuido y paralelo con tolerancia de fallos antes del proceso de reconfiguración dinámica.

	*N*0	*N*1	*N*2	*N*3	*N*4	*N*5
*F*0	0	1	2	3	4	*P*
*F*1	5	6	7	8	*P*	9
*F*2	10	11	12	*P*	13	14
*F*3	15	16	*P*	17	18	19
*F*4	20	*P*	21	22	23	24
*F*5	*P*	25	26	27	28	29

Figura 6. Matriz de un sistema de archivos distribuido y paralelo con tolerancia de fallos después del proceso de reconfiguración dinámica.

Con la incorporación de la reconfiguración dinámica el sistema queda con el siguiente diagrama de estados posibles (fig. 7). Donde:

- **Estado Normal**: Indica que todos los servicios del sistema se encuentran en correcto funcionamiento. Esto quiere decir que no existe ningún nodo que se encuentre con bloques de datos no disponibles debido a un fallo de los discos o de los mismos nodos.

- **Estado Degradado**: Indica que el sistema esta presentando un fallo en algún nodo o disco dentro de un nodo que impide el normal funcionamiento del VRAID. Sin embargo, esto no significa que el sistema no pueda entregar sus servicios, sino que estos tomarán un mayor tiempo debido al necesario cálculo de los datos que no se encuentran explícitos sino que deben ser reconstituidos con los códigos correctores de errores.

- **Estado Reconstrucción**: Este estado indica que existe un nodo o un disco en algún nodo que no permite la normal entrega de los servicios. En este estado el sistema se encuentra con la unidad que provoco el fallo reparada, pero sin los datos que debe contener. En consecuencia el sistema trabaja con un procesos reconstructor que genera y

almacena los datos en la unidad reparada en paralelo a otras solicitudes que le sean hechas que puedes ser respondidas en modalidad degradada o normal según sea el caso

- **Estado Fuera de Servicio**: Este estado indica que el sistema se encuentra fuera de servicio. La ocurrencia de este estado la determina el fallo de más de un nodo o disco dentro un nodo..

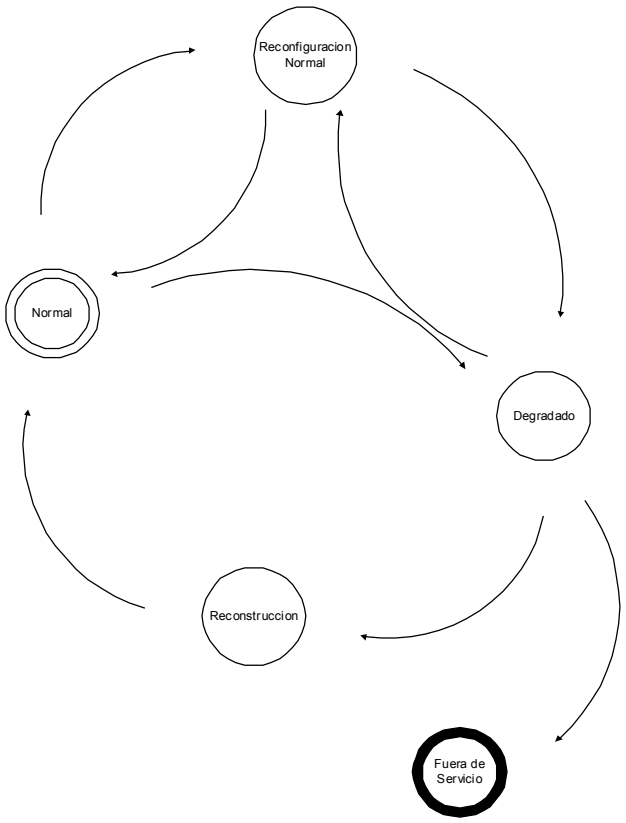


Figura 7. Estados de un sistema de archivos distribuido y paralelo con tolerancia a fallos y reconfiguración dinámica.

Este diagrama muestra en la parte superior el nuevo estado que tendrá el sistema denominado "**Reconfiguración Normal**" cuyo propósito es indicar a cada uno de los componentes que el sistema se esta reconfigurando con la finalidad de incorporar los nuevos nodos. A su vez, este estado de reconfiguración podría estar trabajando tanto en modo normal como en modo degradado (denominandose "**Reconfiguración degradada**"), sin embargo en esta investigación solo se ha trabajado con la reconfiguración en modo Normal.

En base a lo mostrado, se han planteado dos nuevos modelos que permitan implementar la solución al problema.

3.2. Primer Modelo de Reconfiguración Dinámica.

Este primer modelo incorpora una nueva capa de software al sistema de archivos distribuido y paralelo con redundancia de datos (Figura 8).

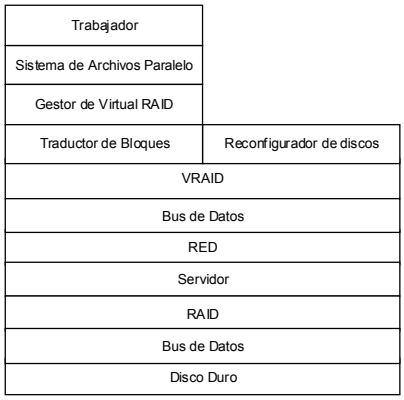


Figura 8. Bloques de un primer modelo de sistema de archivos distribuido y paralelo con tolerancia a fallos y reconfiguración dinámica

Esta nueva capa contiene 2 procesos que trabajan en forma concurrente, el primero de ellos denominado *Traductor de Bloques* se encarga de transformar las solicitudes de lectura/escritura que vienen dirigidas a nodos determinados desde los trabajadores a nuevas solicitudes dirigidas a la nueva configuración del raidmap. El segundo bloque llamado *Reconfigurador de discos*, es un proceso que tiene como misión realizar todas las operaciones necesarias para que el sistema incorpore los nuevos nodos y redistribuya la información. La descripción de los nuevos bloques es la siguiente:

Traductor de Bloques: Este proceso se encarga de redireccionar las solicitudes de entrada/salida provenientes desde los trabajadores. La motivación para crear este proceso viene de la problemática consistente en que el sistema de archivos no debe detener la entrega de los servicios mientras se encuentre en un estado de reconfiguración. A medida que el proceso de reconfiguración va avanzando por la matriz, esta queda particionada en tres zonas. La *primera zona* contiene las franjas de paridad que han sido reconfiguradas y cuya distribución abarca la totalidad de los nodos. La *segunda zona* esta conformada por las franjas de paridad que se encuentran bloqueadas puesto que el proceso reconfigurador esta trabajando sobre ellas. Finalmente la *tercera zona* esta compuesta por las franjas de paridad que aún no han sido reconfiguradas y que todavía se encuentran distribuidas como en un principio.

Debido a que la matriz se encuentra dividida en tres zonas (figura 9), el proceso traductor de direcciones debe ser capaz de discriminar a cual zona va dirigida la solicitud para de esta manera redireccionar la petición. Si la solicitud va dirigida a la zona reconfigurada el traductor de direcciones enviará esta solicitud a una nueva posición que será calculada en base a las expresiones de la ecuación (1).

$$j\_proy = \mathrm{int}\left(\frac{bloque}{n-1}\right)$$

$$i\_proy = \left(bloque\%(n-1)\right) + \mathrm{int}\left(\frac{bloque}{((n-2)*(j\_proy+1))+1+\left(n*\mathrm{int}\left(\frac{j\_proy}{n}\right)\right)}\right) \qquad (1)$$

donde:

n: número de nodos del sistema.

bloque: número del bloque a trasladar [0..(m*(n-1))-1]

j_proy: número de la franja de paridad en la cual será trasladado el bloque [0..m-1].

i_proy: número del nodo en el cual será trasladado el bloque[0..n-1].

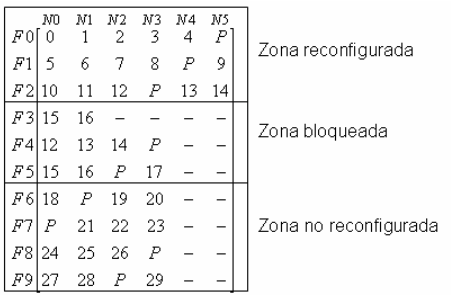


Figura 9. Zonas de reconfiguración.

Si la solicitud va dirigida a la zona bloqueada, el traductor de direcciones se encargará de postergar la solicitud hasta que dicha zona sea reconfigurada, para esto se utiliza un cerrojo de tal forma de proteger esa zona de datos.

Finalmente si la solicitud va dirigida a la zona no reconfigurada, el traductor de direcciones solo se encarga de retransmitir la petición tal cual fue realizada puesto que los bloques aún se encuentran en la misma posición.

Reconfigurador de Discos: Este módulo se encarga de redistribuir la información contenida en los nodos de forma tal que ahora incorpore a los nuevos nodos que se han agregado al sistema. Para esto hace un recorrido por la matriz, bloqueando las franjas de paridad que contienen a los bloques que en ese momento se están reconfigurando, con el fin de que ningún otro proceso lea o escriba en dichos bloques. Un parámetro importante para el rendimiento del sistema durante el funcionamiento de este módulo, es el número de franjas de paridad que deben bloquearse (figura 10) la cual esta determinada por el número de bloques que contendrá el grupo a reconfigurar.

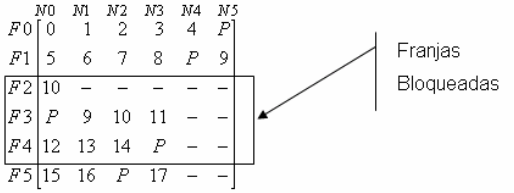


Figura 10. Bloqueo de franjas de paridad.

Independiente del modelo a implementar, existe un problema que debe solucionarse que tiene que ver con el bloqueo del acceso a los bloques. Para esto existen dos alternativas:

1) **Restringir el acceso a bloques específicos**: Esta opción significa que solo se impedirá el acceso los bloques que estén siendo relocalizados en ese momento, lo cual trae como ventaja minimizar la posibilidad que el bloque que se está solicitando desde los trabajadores se encuentre no disponible en ese instante. Sin embargo, esto provoca un aumento en el número de operaciones que debe realizarse y con esto el tiempo que tarda la reconfiguración. Por ejemplo, para mostrar esto de una manera mas sencilla se puede llevar esta situación al extremo, es decir, se puede pensar que el proceso reconfigurador solo irá tomando un bloque y lo llevará a su nueva posición, luego tomará el siguiente y así sucesivamente. De lo anterior se puede ver que cada vez que se relocalize un bloque será necesario calcular nuevamente la paridad tanto de la franja de origen como de la franja de destino, lo que implica un considerable aumento en el número de cálculos necesarios.

2) **Bloquear el acceso a franjas específicas**. Esta opción propone que el bloqueo de acceso abarque un número determinado de franjas completas, produciéndose así un aumento en la probabilidad de encontrar en un determinado momento bloqueado el bloque que se está solicitando desde los trabajadores ya sea para lectura o escritura. Por otro lado, la ventaja de este método se encuentra en que se reduce la cantidad de veces que debe calcularse la paridad a solo una vez por franja, lo que en comparación con la opción anterior en la cual era necesario calcular la paridad una vez por cada bloque de la franja lleva a un menor tiempo de procesamiento.

En base a lo explicado en los puntos anteriores, se decidió utilizar la segunda alternativa puesto que aunque se aumenta la probabilidad de que encontrar bloqueado un bloque solicitado por un trabajador, esta se puede seguir

considerando despreciable debido a que el número total de bloques dentro del sistema de archivos es considerablemente mayor que el número de bloques dentro de una franja de paridad.

Finalmente, para agrupar todos lo conceptos anteriormente descritos, es decir, las franjas de paridad, tamaños de las unidades de paridad, operaciones de bloqueo/desbloqueo, cálculos de paridad, etc., es necesario generar un algoritmo (figura 11) que permita de una manera ordenada y metódica aplicar cada una de estas ideas de una forma conjunta. Para el sistema de archivos distribuido y paralelo con redundancia de datos desarrollado en [6] se tiene el siguiente algoritmo:

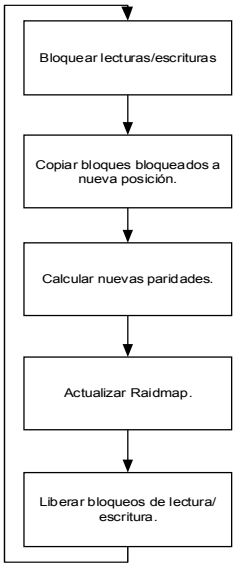


Figura 11. Diagrama de flujos del algoritmo de reconfiguración.

Bloquear lecturas/escrituras: El primer paso bloquea todas las franjas de paridad que serán trasladadas a su nueva posición, así también como las franjas de paridad que se encuentran en las posiciones de destino de los bloques a trasladar.

Copiar bloques bloqueados a nueva posición. Como segundo paso, se copian todos los bloques que pertenecen a la franja de paridad de origen hacia la franja de paridad destino.

Calcular nuevas paridades . El tercer paso es calcular y escribir las unidades de paridad de las franjas de origen y de destino.

Actualizar Traductor de Bloques. El cuarto paso es indicar en el traductor de bloques la nueva posición del puntero de reconfiguración con el fin de que este pueda redireccionar las acciones solicitadas a los bloques de manera correcta.

Liberar bloqueos de lectura/escritura. El quinto paso dentro del proceso de reconfiguración es liberar las franjas de paridad que se encontraban bloqueadas puesto que estas ya se encuentran actualizadas y entregando un normal servicio para los trabajadores que las soliciten.

4. EVALUACIÓN.

Para la evaluación definitiva del modelo propuesto, se procedió a modificar el simulador del sistema de archivos distribuido y paralelo con redundancia de datos desarrollado en [6] para incorporar los nuevos módulos. En primer lugar se detectaron cuales son las librerías sensibles a las nuevas funcionalidades para su posterior modificación. Principalmente, debieron modificarse aquellas secciones que tienen que ver con el funcionamiento de los procesos trabajadores que en este sistema se encuentran aleatoriamente repartidos con tiempo entre llegadas correspondientes a una distribución uniforme.

Las modificaciones realizadas a estos módulos corresponden a la adición de operaciones que les indiquen a los trabajadores que deben detener su ejecución hasta que el bloque sobre el cual deben actuar se encuentre liberado del proceso de reconfiguración. Además fue necesario agregar una funcionalidad, correspondiente al traductor de

bloques, que le indique a los nodos cual es el verdadero bloque sobre el cual debe trabajar. Por otro lado, estas nuevas funcionalidades provocan un retardo que es detectado por el sistema y permite generar las estadísticas finales que se muestran en los gráficos de los próximos capítulos.

4.1. Selección del tamaño de la Unidad de Reparto.

En definitiva, se ha optado trabajar con los siguientes tamaños de unidades de reparto:

Tamaño Gsuperstripe: Este es el tamaño de la unidad de reparto para el gestor de vraids, el cual se ha fijado en 4KB para las cargas OLPT y de 256KB para las cargas Científicas.

Tamaño Supertripe: Esto corresponde al tamaño de la unidad de reparto de las unidades VRAID y se ha fijado en 4KB para las cargas OLPT y de 64KB para las cargas Científicas.

Tamaño Stripe: Esta es unidad de reparto de los RAID que pertenecen a cada uno de los nodos del sistema. Los valores que se han fijado en este caso son de 4KB para las cargas OLPT y de 4KB para las cargas de tipo Científica.

El criterio utilizado para seleccionar los distintos tamaños de las unidades de reparto fue el hecho que es necesario comparar los resultados de este estudio con otros experimentos anteriores realizados en [6] y en [7], estudios en los que se utilizaron los valores anteriormente mencionados. Sin embargo es bueno mencionar que los valores seleccionados en esos estudios garantizaban que la división de la petición del cliente permitía el máximo del paralelismo dado en el sistema[7].

4.2. Selección de la Carga de Trabajo.

Tomando en consideración los estudios realizados por empresas y universidades se ha considerado trabajar con las cargas de tipo transaccional y las de tipo científica ya que representa a un porcentaje alto de la actividad de un Sistema de Archivo Distribuido y Paralelo.

Carga de Trabajo OLPT:

Las principales características de las cargas transaccionales con las que se trabajaron en este estudio se muestran en la Tabla 1.

Tabla 1. Perfil de una carga de trabajo OLPT.

Distribución Probabilística	Uniforme
Lecturas de 4 KB	8%
Escrituras de 4KB	88%
Lecturas de 24KB	2%
Escrituras de 24KB	2%

Carga de Trabajo Científica:

Para la carga de tipo científica se tiene el perfil de la Tabla 2.

Tabla 2. Perfil de una carga de trabajo científica.

Distribución Probabilística	Uniforme
Accesos	50% a 10 archivos de 5MB
Accesos	50% acceso secuencial a 1 archivo de 100MB
Lecturas a los archivos de 5MB	10% lecturas de 5MB
Escrituras a los archivos de 5MB	90% escrituras de 5MB
Lecturas al archivo de 100MB	90% lecturas de 1MB
Lecturas al archivo de 100MB	10% escrituras de 1MB

Al igual que en la selección del tamaño de la unidad de reparto y con el fin de poder realizar una comparación posterior, se ha decidido continuar con la línea planteada en los estudios realizados en [6] y en [7] y se ha optado por generar las pruebas en bases a las cargas de tipo OLPT y las cargas de tipo Científica. Además, se han realizado las pruebas utilizando tiempos entre peticiones generadas por las cargas de trabajo distribuidos de manera uniforme con tiempo de 250, 500, 750, 1000 t 1250 ms. en promedio.

4.3. Selección de la configuración del VRAID.

En cuanto a la configuración del VRAID, las pruebas se han realizado tomando en cuenta que se debe trabajar al igual que en [7], por lo que para las primeras pruebas que tienen que ver con la medición de la productividad en el sistema, se han utilizados configuraciones de dos sistemas VRAID con 3, 5 y 9 nodos tanto para las cargas OLPT como para las de tipo científica. Luego, para las pruebas que tienen que ver con la medición de los tiempos de lectura y escritura, se utilizaron configuraciones de dos VRAID de 5 nodos cada uno.

Conviene decir que para todas las configuraciones generadas para las distintas pruebas se han distribuido los nodos de los VRAID en una red de topología bus de 100Mbps y además cada uno de estos contenía una matriz de discos RAID.

4.4. Selección de la configuración de los RAID.

Para la selección de las configuraciones de las matrices de discos RAID contenidas en cada uno de los nodos, se utilizó el mismo criterio que para las configuraciones anteriores, es decir aquellas que permitiesen ser comparadas con los trabajos realizados en [6] y en [7].

En definitiva, cada una de las matrices RAID estaba compuesta por 5 discos y con una configuración de tipo RAID5 con simetría izquierda, además los discos estaban conectados con un bus de tipo SCSI2 con un ancho de banda de 20 MB/s y concurrencia activada. Resta decir que los discos simulaban discos de marca IBM, modelo ULTRA2 con 3534 cilindros, 21 pistas por cilindro, 110 sectores por pista, 10000 revoluciones por minuto, tiempo mínimo de búsqueda de 2.4 ms, tiempo promedio de búsqueda de 9 ms, tiempo máximo de búsqueda de 19 ms, caché de 2048 KB y 4GB de tamaño.

4.5. Resultados Carga OLPT

Después de ejecutar el sistema simulador con las modificaciones necesarias, se midieron los tiempos de respuesta del sistema para peticiones de lecturas y escrituras encontrándose lo siguiente:

- *Tiempos de Lectura (figura 12):*

En este caso, se graficaron los valores obtenidos para peticiones cuyos tiempos entre llegadas se distribuyen de manera uniforme entre 0 y 250, 500, 750, 1000 y 1250 milisegundos. Se puede apreciar que los valores obtenidos tanto para el modo normal (puntos marcados con cuadrados) como para el modo de reconfiguración (puntos marcados con triángulos), no existe una gran diferencia. Esto se puede explicar debido a que el proceso de reconfiguración adiciona tan solo una pequeña cantidad de proceso que es despreciable si se compara con los tiempos totales del sistema.

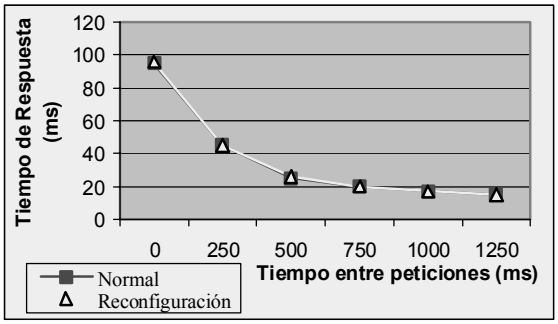


Figura 12. Gráfico de tiempos de lecturas del sistema para el modo normal y en reconfiguración con cargas de tipo OLPT.

- *Tiempos de Escritura (figura 13):*

Al igual que el gráfico anterior, este presenta los valores obtenidos para el sistema en los estados normal y en reconfiguración para los tiempos entre llegadas de las peticiones de lectura/escritura distribuidas de manera uniforme entre 0, 250, 500, 750, 1000 y 1250 milisegundos. De la misma manera, se aprecia que el tiempo de respuesta del modo reconfiguración cuando los tiempos entre llegadas de las peticiones es pequeño es levemente superior a los tiempos del estado normal, tendiendo estos tiempos de respuesta a igualarse a medida que se incrementan los tiempos entre llegadas. Si se toma en cuenta que todo experimento considera un rango de incertidumbre, puede considerarse que la diferencia de los primeros valores es prácticamente nula.

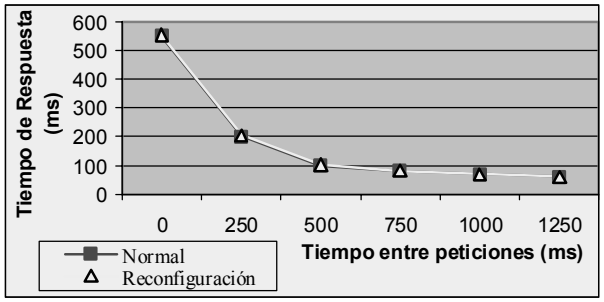


Figura 13. Gráfico de tiempos de escrituras del sistema para el modo normal y en reconfiguración con cargas de tipo OLPT.

4.6. Resultados Carga Científica

Para el caso de las cargas de tipo científica, se realizo el mismo tipo de pruebas que para las cargas de tipo OLPT. Los resultados obtenidos para los distintos tipos de peticiones se muestran en la fig. 1.4

Los tiempos de lectura de las cargas de tipo científica se realizaron con los mismos valores que para las carga de tipo OLPT. En este gráfico se puede apreciar que los tiempo de lectura para el tipo de cargas de tipo científica es mayor que los tiempos de lectura para cargos de tipo OLPT tendiendo a igualarse a medida que aumentan los tiempos entre peticiones. Sin embargo, al igual que en las cargas de tipo OLPT, la diferencia de tiempos entre las modalidades normal y reconfiguración son prácticamente los mismos estos se muestran en la fig. 1.5

Se puede apreciar al igual que el gráfico anterior que los tiempos de respuesta son mejores para las cargas de tipo OLPT que para las de tipo científica sobre todo cuando los tiempos entre llegadas de las peticiones son bajos. Además también se puede ver que de la misma manera que los gráficos anteriores, el tiempo de respuesta del sistema para los modos normal y reconfiguración es bastante similar.

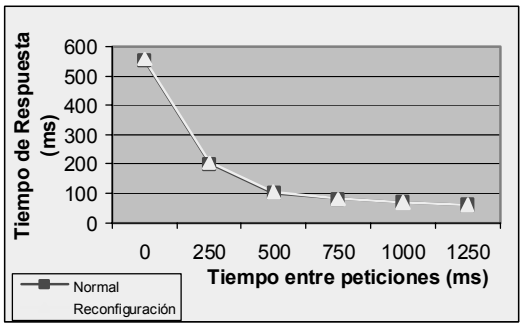


Figura 14. Gráfico de tiempos de lecturas del sistema para el modo normal y en reconfiguración con cargas de científica.

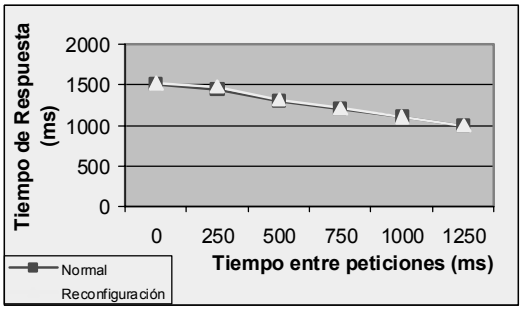


Figura 15. Gráfico de tiempos de escrituras del sistema para el modo normal y en reconfiguración con cargas científica.

5. CONCLUSIONES

Del trabajo desarrollado en esta tesis se puede concluir lo siguiente:

1. A medida que el tiempo entre peticiones aumenta, el sistema tiende a comportarse de la misma manera en modalidades normal y de reconfiguración.

2. El modelo seleccionado para realizar las pruebas es una alternativa viable para ser implementada en la realidad puesto que cumple con el objetivo de reconfigurar el sistema desde un estado inicial de desbalance a uno final en el que los datos se encuentran repartidos de manera homogénea.

3. Cuando el sistema se encuentra en modo de reconfiguración existe una carga de trabajo adicional que se puede considerar despreciable respecto a la carga de trabajo total cuando el sistema se encuentra en modo normal.

Esto se explica por medio del siguiente análisis: En las pruebas realizadas se utilizó un sistema compuesto por 50 discos de 4GB cada uno por lo tanto el tamaño total del sistema es de 208,98 GB. El tamaño de cada franja de paridad es de 640KB, existiendo por lo tanto 326541 franjas. El proceso de reconfiguración bloquea dos franjas, una para la lectura de la franja que se esta trasladando y otra que es la franja que se esta escribiendo. Esto implica que la probabilidad de que una petición se efectúe sobre una franja bloqueada se obtiene en la ecuación (2):

$$P = \frac{2}{326541} = 6{,}12 * 10^{-6} \qquad (2)$$

De aquí se puede apreciar que el valor de P es extremadamente pequeño, sin embargo si se considera que el tiempo entre peticiones es de 0, y el número de peticiones que se ejecutan es de 150 como en el caso de las pruebas realizadas, entonces suponiendo que todas las peticiones han bloqueado una franja la ultima petición tiene una probabilidad igual a la que se encuentra en la ecuación (3) de encontrar una franja ocupada:

$$P = \frac{152}{326541} = 0{,}0004593 \qquad (3)$$

De lo anterior se aprecia que las probabilidades de que una petición encuentre ocupada la franja sobre la que realizará la operación es baja. De todas maneras es bueno hacer notar que estos cálculos suponen que las peticiones son independientes lo que en la realidad no es así puesto que las peticiones tanto de lectura y escritura tienden a agruparse sobre franjas contiguas.

REFERENCIAS

[1] Real Academia Española. Diccionario de la Lengua Española. Pagina 822. Junio de 1992.

[2] D. Patterson, G. Gibson, and R. Katz. A Case for Redundant Arrays of Inexpensive Disks (RAID). *In Proc of ACM SIGMOD* p 109-116. ACM, June 1988.

[3] N. Nieuwejaar, D. Kotz, A. Purakayastha, C.S. Ellis and M. Best, "File access characteristics of parallel scientific workloads" Tech. Rep. PCS-TR95-263, Dartmouth College, Aug. 1995.

[4] D. Kotz and N. Ñieuwejaar. File System Workload on a Scientific Multiprocessor. IEEE Parallel and Distributed Technology. Systems and Aplications, pages 134-154, Spring 1995.

[5] M. Ñelson, B. Welch, and J. Ousterhout. Caching in the Sprite Network File System. *ACM Transactions on Computer Systems*, 6(1):134-154, February 1988.

[6] Rosales 98 Rosales F, Vega R., Achieving Data Availability on Parallel and Distributed File Systems, Proceding 3° International Meeting On vector and Parallel Processing, Facultadade de Engenharia da U. do Porto Portugal Jun 1998, pag 645-650.

[7] Vega[01] Vega R., Romo C., Rosales F., Mejora de la disponibilidad de datos de un subsistema de E/S paralelo mediante agrupación de componentes. XXVII Conf. Latinoamericana de Informática (CLEI 2001), Sep. 2001, U. de Los Andes, Mérida, Venezuela.

[8] Vega[00] Vega R., Rosales F, Carretero J, Propuesta y Evaluación de un Modelo de E/S Redundante para un Sistema de Ficheros Distribuido y Paralelo. XXVI CLEI 2000, Septiembre 2000, Tecnológico de Monterrey, México. Elegido entre 10 mejores artículos

ACQUA: A Conceptual Data Model for Designing and Implementing Databases for Water Resources Management in GIS Environment

Angelo Brayner
University of Fortaleza, Department of Computer Science (MIA), Campus da Unifor, Bloco D,
Fortaleza, Ceará, Brasil, 60811-341,
brayner@unifor.br

and

Joney Rosas Cysne
Federal University of Ceará, Department of Computer Science (MCC), Campus do Pici, Bloco 910,
Fortaleza, Ceará, Brasil, 60455-760,
joney@funceme.br

Abstract

This paper proposes a conceptual data model, called ACQUA, for representing water resources. The proposed model has the capability of capturing the semantics of geographic phenomena, which are related with water resources, such as reservoirs, rivers, channels, pipelines, lakes and singularities (e.g. waterwithdrawal, waterdischarge, confluence, riverhead and monitoring-station). Using ACQUA it is possible to model the complex water resources network, its physical properties and its spatial information, within a geographic context. The model captures and represents information about an area and may be used as support for urban planning, water-related crisis management (such as, water offer and demand during a dry period) or as a basis for scenario simulation.

Keywords: Conceptual Data Models, Spatial Databases, GIS and Geographic Applications.

1. INTRODUCTION

From a database technology standpoint, a Geographic Information Systems (GIS) represents an information system with the capability of capturing the semantic of geographic phenomena, in order to storage them as geographic data in a special type of database, a spatial database. In this sense, it is possible to use database system facilities to manage geographic data.

GIS are used to collect, analyze, and present information of geographic world, describing the physical and logical properties (Shekhar et al. [1]). It may be used to support geographical data from different sources, such as cartographic data, urban and agricultural census data, satellite images, networks and surface numerical models. Viewed as a system composed by hardware, software, collection of spatial and non-spatial data and human resources (Nyerges [2]), a GIS internally presents hierarchic components distributed in layers. These layers are called *(i)* user interface, *(ii)* data input and integration, *(iii)* vector processing and raster functions, *(iv)* viewing and plotting functions, *(v)* storage and accessing data functions.

Geographical world is represented in two perspectives: field model and object model. The field model see the world like a continuous surface, and the geographical phenomena (fields) usually are represented by tessellation or matrix structures. At object model, the world is represented like a surface occupied by identified objects, which can be modeled by three basic structures: points, lines and polygons (Câmara et al. [3]). Geo-referenced data may present three distinct components: a non-spatial component which describes the phenomenon; a spatial component which describes spatial position of the phenomenon, capturing geometric and topological properties, and a time component which represents time properties of the geographic data (Silberschatz et al. [4]).

Over the last decades, water related problem has experienced meaningful changes. Nowadays, water is a scarce natural resource in worldwide scale, and endowed of economic value (Campos and Studart [5], Souza [6]). Of course, managing efficiently water resources has become a challenge. Some researchers have proposed data models for representing geographic phenomena in spatial databases to support an efficient management of environmental resources. By doing this, it is possible to use spatial database functionalities for managing water related data.

According to Borges et al [7], a geographical data model should provide *(i)* a high level of abstraction to represent geo-fields and geo-objects *(ii) a r*epresentation for all kinds of data and spatial relationships involved in geographical applications, *(iii)* definition for spatial integrity constraints, *(iv) s*upport to represent geo-classes or non geo-classes, *(v)* multiple visions for the same object, *(vi) i*ndependence between conceptual data model and physical data model.

In this sense, this paper proposes a model, called ACQUA, for representing water resources. The proposed model provides the necessary support to represent a complex water resources network, capturing hydrological and hydrographic information, and the geometry and topology of the geo-referenced objects belonging to the network. Using the ACQUA model one can represent water-resource related information about a given area allowing: identification of all sections of a given channel, pipeline or river in order to locate where hydraulic structures are installed for irrigation, for example. Furthermore, it is possible to determine the percentage of irrigated areas, to control water offer and demand during a dry period and to register data collect by monitoring stations. It is important to note that the model is able to capture information (spatial and non-spatial semantics) of any other water related geographic phenomena such as lake, well, reservoir, dam, monitoring-station, waterwithdrawal, waterdischarge, etc.

This paper is organized as follows. In the next section, we review some GIS conceptual models using different approaches. In section 3, we motivate the applicability and feasibility of our proposal by describing an example that shows how the ACQUA model organizes information about an area. In section 4, we present the core of the model to explain its coverage. In section 5, we presents a brief report about ACQUA VIEW tool, designed and implemented to manage water resources, based on concepts and principles of the ACQUA model. Section 6 discusses the applicability of our proposal. Finally, section 7 concludes the paper.

2. RELATED WORK

A conceptual spatial data model for representing geographical data poses challenges to database designers. In order to develop conceptual spatial data models, some approaches, methodologies and techniques have been used to support and to understand the complex GIS world. In general, the models are developed following the entity-relationship or object-oriented approach. Data models for representing geographic phenomena in GIS can be categorized into field-based models and object-based models or both. Each GIS data model, in particular way, can be considered an innovate model, according with its extensions, features, perspectives and applications (Erwig et al.[8], Patel et al.[9], Jones and Ware[10], Balaguer and Gordillo[11], Laender et al.[12]).

Many researches have proposed spatial data models based on the object-oriented approach. PDM (Dayal and Manola [13], Manola and Orenstein [14]), OO Spatial Model (Nabil and Subramanian [15]) and conceptual model used by SPRING GIS (Câmara et al. [3]) are some attempts to model GIS using object-oriented approach.

The PDM (Dayal and Manola [13], Manola and Orenstein [14]) - Probe Data Model - provides two basic types, entities and functions, to represent data objects. The model also provides PDM algebra to perform manipulation of PDM databases, including formation of generalization hierarchies, new entities types and multi arguments functions. The PDM allows several representations of a given spatial object, depending on the user perspective (different schemas) and takes the approach of modeling topological relationships such as "adjacent-to" explicitly.

In Nabil and Subramanian [15], the OO Spatial Model is based in a combination of data-driven approach (i.e. concentrate in the attributes of the objects) and responsibility-driven approach (i.e. concentrate in the functions or responsibilities of the objects) enhancing the ideas presented in PDM model. PDM was improved by notion of user perspective related with objects representation and implementation of a class of imprecise spatial operators such as "close-to", "between" and "adjacent-to". The base classes of the model are Spatial-feature (which is an abstract class that represents a template for spatial feature) and Line-segment (which is a concrete class that represents low-level spatial objects).

The model proposed for the SPRING GIS (Câmara et al. [3]) represents geo-world in three classes: Non-spatial, Geo-object and Information-plan class. The Information-plan class is a generalization of Geo-field class and Register-class. The Geo-field class and Geo-object class are the roots to Geo-classes basic hierarchy.

On the other hand, some researches have followed the entity-relationship approach where the data are described as entities, relationships and attributes. GISER (Shekhar et al. [1]) and "Roads" data model (Barnett and Carlis [16]) are examples of that approach.

GISER (Shekhar et al. [1]) - Geographic Information System Entity Relational - unifies the field-based and object-based approaches, and explicitly represents the discretization aspects. GISER attempts to support the entire GIS process, from the input of data measured and discretized to the display of this entity and all the processing that must be performed. GISER has also extended entity-relationship model to include continuous fields (EER).

It is important to emphasize that several data models for GIS were developed to represent geo-referenced phenomena by providing the support to associate "meaning" to a spatial object. In the other words, the users have to understand, to refine and to represent the semantics of the geographic phenomena and their relationships. For example, the SPRING GIS model provides the elements to model a spatial object. However, the user has to associate meanings (semantic) to this object. That is, the model provides the support to represent a sequence of lines. None of less the user has to associate a "meaning" to that sequence of lines in order to capture the information that the lines represents a given river, for example.

The "Roads" data model (Barnett and Carlis [16]) is an example of a data model driven for a specific domain. It addresses the problem of cartographic generalization or map generalization in the context of a common map features called "Roads". Complex crossroads, highways, streets, interstate lines etc, are represented in the model. Moreover, the model includes specific types of data to represent explicitly the rules the will control the desired aesthetics of the objects at digital representation. Route, Partitioning, Section and Route-rule are the base entities of the model. Observe that in this model the spatial objects have already a meaning (e.g. road, route and section).

3. MOTIVATING EXAMPLE

We motivate the applicability and feasibility of our proposal by describing an example modeled according to the reality of a semi-arid region. The example will analyze an imaginary river, called Mandacaru River, although describing real structures. The idea is to understand a river as sequence of sections, where each section represents a stretch of the river. Each section of Mandacaru River has begin and end points. Each point is treated as a singularity point, which may have spatial and non-spatial attributes. Suppose that the Mandacaru River has the sections presented in Figure 1.

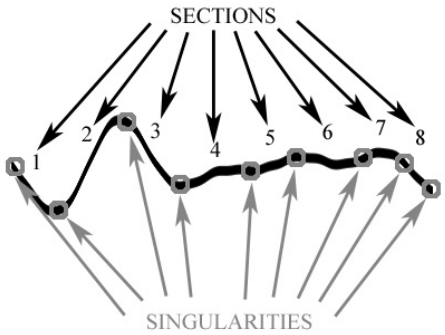


Figure 1 - Mandacaru River Topology.

The identification of each singularity helps to understand and to analyze the water related problems of each stretch of the river. It is interesting to note that a singularity which represents the end point of a section may also represents the begin point of the next section. The Mandacaru River sections (with begin and end points) are described in Table 1.

It is important to observe that a singularity may be a link-point to another section or another water network as a channel, pipeline, reservoir set, well set or another river. For example, the same singularity, called inflect point, represents the end point of the section 1 and the begin point of the section 2. The singularity inflect point may represents a sharp curve, enlargement, reduction, elevation or lowering, in a certain section in the pipe, channel or river. There is also the singularity waterwithdrawal. Waterwithdrawal are important hydraulic structures built to transfer water (e.g. using water pumps) to another hydraulic structure, such as a pipeline or channel. This channel or pipeline conducts the water to others reservoirs, wells, channels and pipelines, which can be water related objects belonging to another water resources network, that is, they do not belong to the water resources network of the Mandacaru River. In general, unfortunately dirty water, without treatment, come back to the rivers through others hydraulic structures, called waterdischarges (or sewers). For this reason, monitoring stations are installed to control water quality. Each river that reaches a principal river represents a confluence. Finally, at the river course, that begun with the singularity riverhead, there is a dam and an island before it reaches the rivermouth.

Table 1 - Mandacaru River Sections.

Begin Point (Singularity)	**Section Number**	**End Point (Singularity)**
Riverhead	1	Inflect Point
Inflect Point	2	WaterWithdrawal
WaterWithdrawal	3	Sewer
Sewer	4	Monitoring Station
Monitoring Station	5	Confluence
Confluence	6	Dam
Dam	7	Island
Island	8	Rivermouth

4. THE ACQUA MODEL

In this section, we describe the ACQUA model using the Unified Modeling Language (UML). UML is a pattern language adopted by OMG (Object Management Group) to describe a conceptual model, in a uniform way. Besides, as UML is an extensible language, it allows introducing new stereotypes for specific applications, if it is needed. Furthermore, the UML formal concepts provide appropriated support for representing the complex objects related with complex relationships that are inherent to the geographic information systems.

The ACQUA Model is focused on superficial water resources and adopts the object modeling, instead of thematic modeling, to represent water world. The proposed model has the following properties: *(i)* clear and simple representation of the objects, *(ii)* capability of capturing the semantics of any real-world object involved in a complex water resources network, *(iii)* independence between conceptual model and physical model, *(iv)* representation of geo-referenced and conventional classes, *(v)* representation of topological relationships.

Four classes compose the core of the proposed model. These classes are called *Natural_Hydrography*, *Water_Reserve*, *Water_Transfe*r and *Singularity* class. Figure 2 depicts the class diagram of the ACQUA model.

The *Natural_Hydrography* class represents water resources sculptured by the nature. This class has two subclasses: *Lake* and *River* class. The *Water_Reserve* class is used to represent hydraulic structures built to storage water for many purposes, such as irrigation, energy supply, human consumption etc. The *Water_Reserve* class has two subclasses: *Well* and *Reservoir* class. The *Water_Transfer* class models hydraulic structures used to transport water from one place to another and it also has two subclasses, called *Channel* and *Pipeline* class. Three distinct subclasses represent sections of rivers, channels and pipelines. Others water-related structures can be modeled through the *Singularity* class.

The *Singularity* class has these subclasses: *WaterDischarge*, *Riverhead*, *Rivermouth*, *WaterWithdrawal*, *Confluence*, *Inflect_Point, Island*, *Monitoring_Station*, *Dam* and *Hydroelectric_Plant* class. These subclasses that represent hydraulic structures or geographic phenomena contribute directly to water system operation, and some of these objects can be relating (linking) the others three classes. For example, the water that emerges from a riverhead (first singularity identified in a river) follows freely until may be repressed by a dam (another type of singularity). In such a case, a great reservoir is created (reservoir class). The water follows its course until it reaches the rivermouth (the last singularity of a river), but before this, others rivers may reach the principal river (confluence singularity) and many waterworks may transfer water from the river to channels or pipelines to supply others places (waterwithdrawal singularity). Note that without singularities the water network topology would be simplified, hiding important details necessary for water resources management.

The *Monitoring_Station* subclass represents control-points of the monitoring process in the water resources network. The inner sensors are used to identify stations type and the exact location where the stations have to be placed. Data input includes precipitation, evaporation, infiltration etc, and are used in conjunction with others measured data for testing alternative monitoring designs and management options. In such case, the *Monitoring_Station* subclass can be specialized in channel station, pipeline station, well station, reservoir station, climatic station or fluvial station. They provide support to certain activities such as water offers evaluation, water demand control, hydro-environmental monitoring (parameters like rivers flow, pluviometry etc), water quality analysis, operational information about hydraulic systems etc. Accordingly, the model will be used to help guide decisions on further evaluating and refining the monitoring network, and evaluating and refining action criteria.

The *Riverhead, Rivermouth, Confluence* and *Island* are subclasses to represent specific natural phenomena of rivers. *Inflect_Point* subclass represents great depressions, elevations, turnings, enlargements, reductions, waterfalls, etc, that change deeply the water way in a pipeline, channel or river. The *WaterWithdrawal* subclass can represent a hydraulic apparatus, or a system of works or fixtures, by which a supply of water is furnished for useful purposes, including dams, sluices, pumps, aqueducts, distributing pipes, etc. Another important hydraulic structure is represented by the *WaterDischarge* subclass. By means of that subclass one can represent the flow of waste and dirty water through the network. Finally, the *Dam* and *Hydroelectric_Plant* subclass that has special functions in the reservoir and river simultaneously.

Two important water-related areas are represented in the ACQUA Model and linked to *WaterWithdrawal* class and *Monitoring_Station* class. The first area associated to the *WaterWithdrawal* class represents the water offer for several purposes, for example industry, psiculture, concessionary, tourism and irrigation. The human subsistence, environmental preservation, service and industry development are influenced directly by water availability. The water offer has direct effects on water management, especially if we assume that the water demand is always higher than water offer. The situation is still more critical in semi-arid regions. By using the ACQUA model, it is possible to represent the main users of water resources in a given region. This can be done by means of the *User* class hierarchy. For example, it is possible to represent who has received grant to use water for irrigation. The second area linked to *Monitoring_Station* class deals with water quality problem. Freshwater demand is increasing rapidly, due to population growth and economic development. For that reason, the analysis of the quality is necessary to control water pureness and environment changes. The quality campaigns with its analysis and samples, measured according specific parameters, are represented by *Campaign*, *Analysis*, *Sample* and *Parameters* classes.

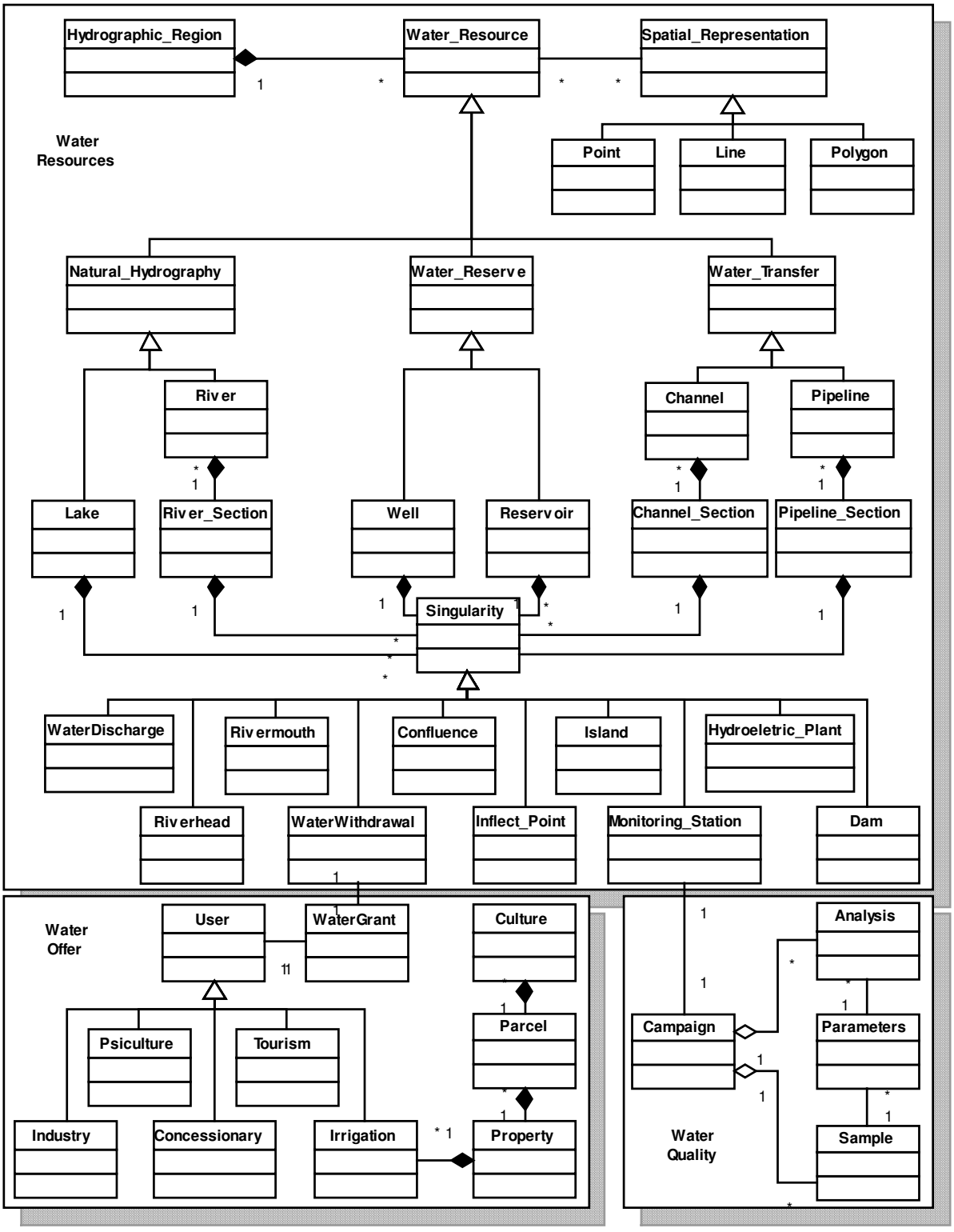


Figure 2 - The class diagram of the ACQUA model.

5. ACQUA VIEW TOOL

To demonstrate the applicability of our proposal it has been implemented a tool, called ACQUA VIEW, which gives the necessary support for managing water resources in a GIS environment. Developed on a Windows platform and using Delphi 5 components, the source code size is approximately 1.45MB. Initially, a database with water-related objects was created in ORACLE v.9i, based on the ACQUA model. Those objects represent water resources of the Brazilian State called Ceará.

The ACQUA VIEW tool has been used to explore the surface water world in an interactive form. This tool offers to final users the following functionalities: interactive analyses on georeferenced maps, "ad hoc" queries, spatial functions, graphical queries and a complete list of all attributes for each object selected on maps.

The query module offers pre-defined queries and SQL commands available through buttons, besides a complete list of tables and fields accessible by all users. The spatial functions provided by the tool answer questions like: *(i)* What are the dams or monitoring stations that we can find in a specific hydrographic region? *(ii)* What is the hydrographic region that a reservoir belongs to? *(iii)* What is the approximate distance in meters, between point A and point B, appointed on map? (See an example in Figure 3) *(iv)* What are the pluviometers found in a specific action ray? (See an example in Figure 4) *(v)* What are the singularities found in a stretch "n" of river X?.

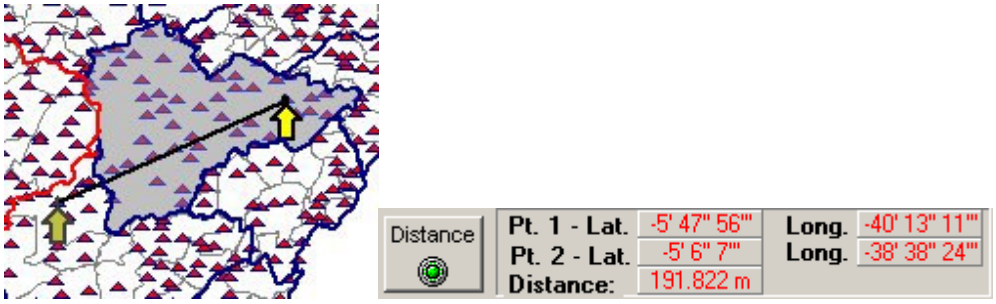


Figure 3 - Spatial Distance Function.

Figure 4 - Ray Action Spatial Function.

For a more accurate representation, evaluation and analyses, all available maps have a zoom function, up to 40x. The map is presented in Figure 5. Furthermore, when a reservoir is selected on map, for example, a short window shows the water mirror of the selected object, detailing its features (See Figure 6). Another panel shows dynamically the names of region, object and city focused, and the approximate localization (latitude and longitude) (See Figure 7). Specific colors are designated for selected regions, selectionable regions, regions in focus and cities, but can be modified by the users during running application.

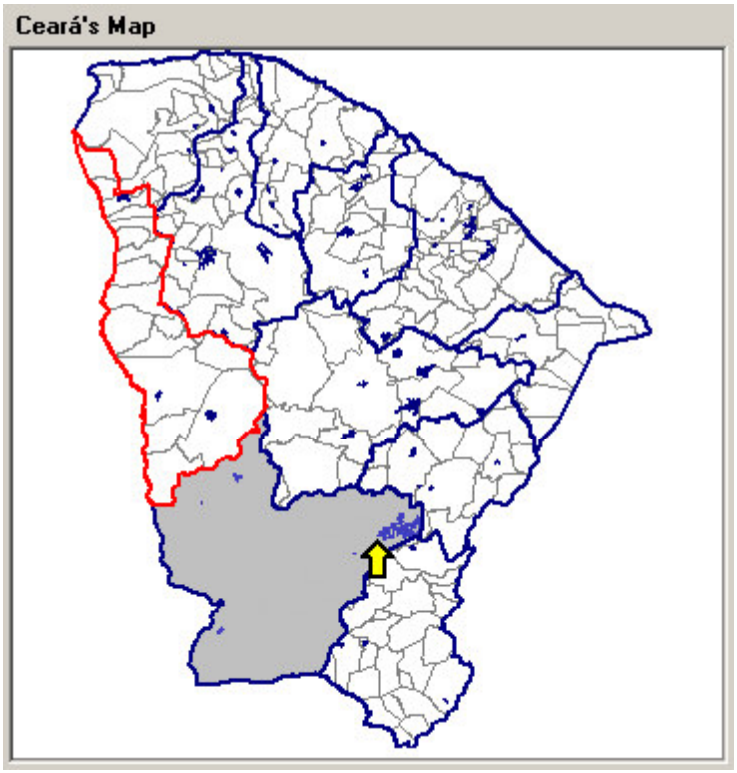


Figure 5 - Main Map.

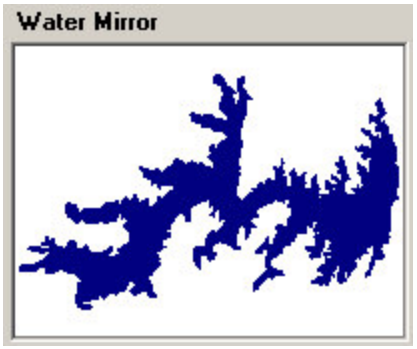


Figure 6 - Water Mirror Map.

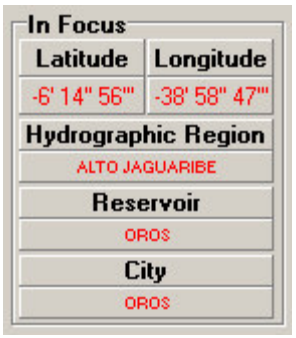


Figure 7 - Information Panel.

6. DISCUSSION

Except "Roads" model (that presents map generalization rules and appropriate entities for representing traffic objects), the other models for modeling geo-referenced phenomena investigated are for general purposes. A data model for general purposes transfers to final users the design tasks related with identification, organization and storage of the geographic phenomena. On the other hand, a data model specific to a given area (e.g. water resources) models geographic phenomena more efficiently since it associates a meaning (semantic) to the phenomena. The gist of the ACQUA model is to capture and represent, all water resources involved in a water network (with its geometry, topology and properties).

In the models with generic representation (geo-objects), the user has to associate "meanings" (semantic) to spatial objects. In the diagram class (Figure 2), observe that the mapping class is almost intuitive when we use the ACQUA model. We will come back to Mandacaru River example (Figure 1 and Table 1) to demonstrate how the ACQUA model represents this geographic phenomenon.

In order to model the water resources network related to the Mandacaru River, an object representing the Mandacaru River is created in the River subclass, a subtype of Natural_Hidrography class and also a subtype of Water_Resource class. For each stretch of Mandacaru River is created an instance in the River_Section subclass, and for each begin and end point an instance is created in the specific subclass of the Singularity class, if it was not created yet. If you have already created an instance, only the relationships are created. Observe that the topological relationships of the water resources network of Mandacaru River are represented in the model. For example, the Dam (river sections 6 and 7) and the Island (river sections 7 and 8) are singularities identified along the course of the river.

Singularities which represent begin or end points, discovered inside or at river's margins, and its topological relationships, add details to the model and help to represent a structure nearest the real structure of the water resources network of the river. The same concept is adopted to represent a channel or pipeline network and its singularities.

As we can see the mapping class is simple and represents each object with its geometry and properties, but also considering its relationships, necessary to understand each phenomenon.

7. CONCLUSIONS

Nowadays, water is a scarce natural resource in worldwide scale. Freshwater problems are acute and worsening. Most arise from the poor management of water resources, lack of financial resources required for sustainable development and efficient utilization of resources, absence of effective regional and basin development plans and shared management, and under-estimation of the groundwater potential to supplement irrigation and drinking water supplies. A GIS should be used to provide support for an efficient water resource management. For that it is necessary providing models for capturing semantics of water resource related geographic phenomena. In this paper, we propose a data model, denoted ACQUA, for representing water-related geographic phenomena, such as lakes, wells, reservoirs, rivers, channels, pipelines and singularities.

ACQUA provides the necessary elements for modeling the complex water resources network, its physical properties and its spatial information, within a geographic context. ACQUA model also provides the information to support scenario simulation for operation of hydraulic systems and to analyze the impact of new hydraulic structures in a given hydrographic region

The proposed model has the capability to produce key information for water resource management, such as:

- The graphical representation of all the water resources of a given hydrographic region;
- How many and what irrigators have grant to use water of a certain resource;
- The water flow required for ensuring the water demand for a given region;
- The water volume that comes in a given water resource;
- The number of wells registered in a given hydrographic region;
- The consumption of water in a planted hectare for a given culture, in a given city, applying a certain irrigation method;
- The behavior between the planned and real water flow in a given year for a certain dam;
- The spatial distribution of monitored wells;
- The impact of rain incident in a hydrographic region;
- The water volume evolution of reservoirs.

Object-relational DBMSs provide the benefits of the object-oriented paradigm (for example, encapsulation, inheritance, methods and polymorphism) and the best capabilities of relational database technology into a single engine (Stonebraker and Brown [17]). For that reason, we have decided for implementing the ACQUA model using the OR database technology.

Finally, the ACQUA model has been used at FUNCEME, a governmental foundation of the Ceará State (northeast of Brazil), with recognized services given for the community in water-related problems. A database has been designed and developed at FUNCEME, using the principles of the ACQUA model to storage the information about the water resources of the Ceará State.

References

[1] Shekhar S., Coyle M, Goyal B, Liu D, Sarkar S. Data Models in Geographic Information Systems. *Communications of the ACM*, April 1997, vol 40, n.4.

[2] T. Nyerges. "Understanding the Scope of GIS: Its Relationship to Environmental Modeling", chapter 8 in Environmental Modeling with Geographic Information Systems, edited by Michael Goodchild, Louis Steyaert, and Bradford Parks. London: Oxford University Press, pp. 75-93, 1993.

[3] Câmara, G., Casanova, M., Hemerly, A. S. Geographic Information Systems Anatomy. Campinas: Computer Institute , UNICAMP, 1996. xi, 197 p. li. (In Portuguese).

[4] Silberschatz, A.; Korth H.F.; Sudarshan S. Database System Concepts. McGraw-Hill, 1996.

[5] Campos, N.; Studart, T. Water Management: Principles e Practices. Porto Alegre: ABRH, 2001 (In Portuguese).

[6] Souza Filho, F.de A.. Water Resources Management Experiences. Environment Ministry (MMA) and Water National Agency (ANA), November, 2001 (In Portuguese).

[7] Borges, K. A. V., Davis Jr., C. A., Laender, A. H. F. OMT-G: An Object-Oriented Data Model for Geographic Applications. *GeoInformatica* 5(3):221-260, 2001

[8] Erwig M., Guting R. H., Schneider M., Vazirgiannis. M. Abstract and Discrete Modeling of Spatio-Temporal Data Types. *Proc. of ACM GIS*, pages 131-136, November 1998.

[9] Patel J., Yu J., Kabra N., Tufte K., Nag B., Burger J., Hall N., Ramasamy K., Lueder R., Ellmann C., Kupsch J., Guo S., Larson J., DeWitt D., Naughton J. Building a Scalable Geo-Spatial DBMS: Technology, Implementation, and Evaluation. *Proc. of ACM-SIGMOD*, pages 337-347, 1997.

[10] Jones C.B., Ware J.M. A Data Model for Representing Geological Surfaces. *Proc. of ACM-SIGMOD*, pages 20-23, 1997.

[11] Balaguer F., Gordillo S. Refining an object-oriented GIS design model: Topologies and Field Data. Proc. of ACM-GIS, pages 56-61, November 1998.

[12] Laender A. H. F., Davis Jr. C. A., Borges K. A. V. Spatial Data Integrity Constraints in Object Oriented Geographic Data Modeling. Proc. of ACM-GIS, pages 1-6, November 1999.

[13] Dayal U., Manola F. PDM: An Object-Oriented Data Model. *Proceedings of the 1986 IEEE Data Engineering Conference*, pages 18-25, 1986.

[14] Manola F., Orenstein J. A. Toward a General Spatial Data Model for an Object-Oriented DBMS. *Proceedings of the Twelfth International Conference on Very Large Databases*, pages 328-335, August 1986.

[15] Nabil R. A., Subramanian R. The Design and Implementation of an Expert Object-Oriented Geographic Information System. *Proc. of ACM-CIKM*, pages 537-546, November 1993.

[16] Barnett L., Carlis J. V. A "Roads" Data Model: A Necessary Component for Feature-Based Map Generalization. *Proc. of ACM-GIS*, pages 58-67, November 1997.

[17] Stonebraker, M., Brown, P. Object-relational DBMSs: Tracking the next great wave. Morgan-Kaufmann. 1999.

Simulación del Proceso de Compra de Artículos en un Mercado Virtual con Agentes BDI

Oscar Pacheco Calcín, Fabio Y. Okuyama y Aurelio M. Dias

Programa de Pós-Graduación em Computación
Universidad Federal de Rio Grande do Sul (UFRGS)
Porto Alegre, Brasil.
{ogpcalcin, okuyama, amdias}@inf.ufrgs.br

Abstract

This work presents the study, analysis and development of a multi agent system whose objective is the simulation of the process of decision making in the purchase and sale of products in a virtual environment. The consumer decisions are based on the characteristcs of the products, as well as in the seller's reputation. The agents of this simulation were developed using AgentSpeak(L), language based on the BDI model. The reason of this choice was it discusses along the work.
Keywords: Multiagent System, Simulation, BDI Agents, Model Consumer..

Resumen

Este trabajo presenta el estudio, analisis y desarrollo de um sistema multiagente cuyo objetivo es la simulación del proceso de toma de decisiones en la compra y venta de productos en un ambiente virtual. Las decisiones de compra son basadas en la caracteristicas de los productos, asi como en el grado de confianza que tengamos en los vendedores. Esta simulación fue desarrollada usando el modelo BDI para la arquitectura interna de los agentes. La razon del porque se escogio este modelo se discute a lo largo del trabajo.
Palabras claves: Sistema Multiagente, Simulación, Agentes BDI, Modelo Consumidor.

1. INTRODUCCIÓN

En ciencias como la administrativa y el marketing, se busca obtener modelos de consumidor los cuales puedan reflejar la forma en la que este toma sus decisiones en el momento de una compra (proceso de compra). Tales modelos, toman en consideración diferentes aspectos, como la influencia que ejerce el ambiente en el consumidor, diferencias entre los consumidores (factores pre - venta), grado de satisfacción o insatisfacción asociado a la compra (factores post venta), etc. Con el modelo, se pueden realizar algunos estudios basados en él, como la simulación de los impactos en las variaciones de los precios y como reaccionan los compradores ante esto, técnicas de marketing, etc. Este tipo de simulaciones, consideradas como estudios de simulación social, han venido siendo foco de atención para la comunidad de Inteligencia Artificial, específicamente, al área de sistemas multiagente, y han abierto la posibilidad de la utilización de agentes como medio para llevar a cabo tales simulaciones.

El propósito de este trabajo es realizar la simulación del proceso de compra, utilizando agentes inteligentes y el modelo propuesto por Engel, Blackwell y Miniard [1], ya que en él se consideran los diferentes estados mentales por los que pasa el consumidor para decidirse por la compra. Los agentes a utilizar son del tipo BDI, debido precisamente a que ellos llevan en consideración los estados mentales como parte de su estructura interna y como forma de representar el conocimiento.

El modelo de consumidor y su relación con la arquitectura BDI se describe en la sección 2. En la sección 3 describiremos la simulación y como los agentes interactúan en el ambiente. En la sección 4 revisaremos los agentes y su estructura interna, ademas de explicar lar herramientas utilizadas para su programación. Los resultados, conclusiones y trabajos futuros serán expuestos en las secciones 5, 6 y 7.

2. TOMA DE DESICIONES EN LOS CONSUMIDORES

De acuerdo com Engel, Blackwell y Miniard [1], el proceso de compra se inicia con el reconocimiento de una ***necesidad***. Esta ***necesidad*** va a desencadenar en el consumidor una serie de acciones con el propósito de satisfazerla (buscar vendedores, consultar características y precios, compararlos y finalmente comprar). Esta ***necesidad*** puede estar representada como un ***deseo*** de compra, ***deseo*** que muchas veces puede tener partes contradictorias entre si. Por ejemplo, podemos tener el deseo de comprar tres articulos, pero el dinero es suficiente solo para dos, por lo que el deseo de compra de un tercero artículo se opone a la compra del resto. Este ***deseo*** constituye el primer estado mental del comprador.

Volviendo a la ***necesidad***, esta puede ser influenciada por tres factores determinantes:

-) Información que posee el consumidor: producto de experiencias anteriores (compras anteriores).
-) Influencias del ambiente: aspectos culturales, familiares, información adquirida de otros compradores, etc.
-) Diferencias individuales entre consumidores: recursos economicos, motivación personal, personalidad, estilo de vida, etc.

Estos tres factores, constituyen elementos informativos en el consumidor, que serán almacenados en su memoria como hechos o ***creencias*** y forman el segundo estado mental.. Con esta información ***(creencias)***, realizamos una busqueda interna y externa para realizar la compra. La busqueda interna la realizamos en nuestra memoria, con el objetivo de reconocer si tenemos o no información que nos pueda ayudar a decidirnos por un vendedor, o por un artículo, etc. Esta información (que también son creencias) y un criterio de evaluación (precio, calidad, satisfaccíon) permitirán decidirse por repetir las características de una compra anterior que pueda satisfazer la necesidad actual, o buscar nuevas informaciones en el ambiente. En caso de escoger la primera acción, estaremos creando un aspecto fundamental en el proceso de compra, la ***fidelidad*** a un artículo o vendedor.

La busqueda externa, es influenciada por el ambiente y las diferencias individuales de cada consumidor. Por ejemplo, el proceso de escoger entre articulos con atributos similares, pero con precios diferentes, es directamente influenciado por la cantidad de dinero que podemos gastar, nuestra personalidad, etc.

La figura 1, muestra estas dos busquedas:

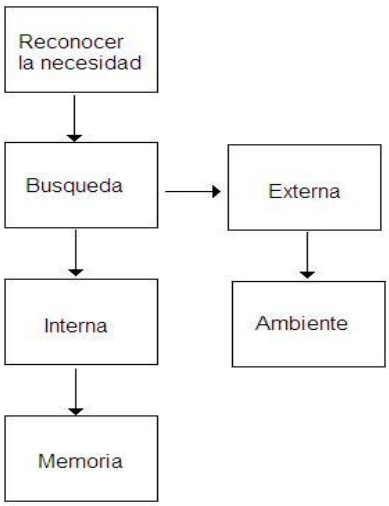


Figura 1. Busqueda interna e externa para el proceso de compra

Las informaciones producto de la busqueda interna y externa, tambien iran almacenandose por el comprador en forma de hechos o *creencias*, que irán modificandose a lo largo del tiempo.

Como etapa final del proceso, despues de evaluar beneficios y desventajas de cada artículo y vendedor, realizamos un tipo de compromiso interno, que nos vincula a la compra de los productos previamente evaluados. Este compromiso interno es tambien llamado de *intención* de compra.

Una vez comprado el producto, podemos realizar una evaluación interna de la compra, y de esta forma modificar nuestras *creencias* sobre determinado producto o vendedor, que a largo plazo, influenciarán futuras compras. De esta forma, en caso de surgir una nueva necesidad, empezamos nuevamente el ciclo descrito.

Resumiendo, en este proceso, el estado interno del consumidor esta representado por tres estados mentales. Ellos son:

-) Las *creencias*, como elementos informativos del consumidor.
-) Los *deseos*, como elementos que desencadenaran acciones a seguir.
-) Las *intenciones*, como compromisos internos no contradictorios.

Y son estos tres tipos de estados mentales los que constituyen los estados mentales de los agentes BDI. Los agentes BDI, propuestos por Georgeff y Rao [7], son agentes cognitivos que basan sus acciones en estos tres estados mentales. Esta arquitectura tiene su base en los estudios realizados por Bratman sobre intenciones, planos y razonamiento práctico [1][2] y han venido siendo utilizados en muchas aplicaciones [8][9]. Una descripción detallada sobre los agentes BDI puede ser encontrada en [6].

En este trabajo tomaremos en cuenta algunos aspectos claves del modelo descrito y los utilizaremos para analizar el comportamiento de los agentes compradores en situaciones que se presentan muy a menudo en las compras. En particular, los agentes compradores estarán en un ambiente donde los vendedores no siempre venden sus artículos con la calidad ofrecida, es decir, ocurre una sobrevaloración de sus productos. Este problema es analizado desde el punto de vista teórico en Mowen y Minor [4].

El punto clave para un correcto desempeño de los agentes compradores en esta simulación, es su capacidad de aprender, ya sea por medio de sus propias experiencias o por medio de la experiencia de los demas. Este aprendizaje envuelve aspectos relacionados básicamente a la modificación de creencias sobre los grados de confianza que todo agente comprador tiene sobre cada uno de los agentes vendedores. A mayor grado de confianza, mayor son las espectativas de que la compra sea exitosa. Un grado de confianza bajo, puede determinar la desconsideración de ese vendedor para futuras transacciones.

3. DESCRIPCIÓN DE LA SIMULACIÓN

A simulación estará compuesta por dos elementos básicos: Los agentes (compradores y vendedores) y nuestro ambiente. De acuerdo a lo explicado en la sección anterior, los agentes compradores necesitan información sobre que artículos comprar, sus características (calidad, precio), que vendedor ofrece ese artículo, y su grado de confianza. El dispondrá de otras informaciones que serán detalladas mas adelante. Estas informaciones forman parte de la base de creencias del agente.

Por el lado del vendedor, este iniciará la simulación con una cantidad de dinero que "cambiara" por articulos para vender. Cada articulo tiene como atributos, ademas del nombre y precio, dos atributo numérico llamados ***calidad_real*** y ***calidad_para_venta*** . El proposito de estos dos atributos es el siguiente: El agente vendedor puede ofrecer un producto con una alta ***calidad_para_venta***, pero en realidad su ***calidad_real*** es menor, de esta forma estará sobrevalorando su producto con la finalidad de intentar venderlo, situación muy común en procesos de compra y venta [4]. El comprador solo conocerá la ***calidad_real*** al comprar el producto. Si se detecta fraude, el grado de confianza en relación al agente vendedor disminuirá. Repetidas disminuciones de grado confianza llevarán a evitar comprar articulos de ese vendedor, aun cuando el precio pueda ser muy bueno

Por el lado del ambiente, este tiene dimensiones de 10x10, toroidal. En cada celda del ambiente, podrá encontrarse uno o mas agentes compradores, una tienda o simplemente estar vacia. En la tienda se encuentra un agente vendedor, y esta puede ser accesada por mas de un agente comprador. Otro elemento presente en el ambiente, son las denominadas "casas" de los compradores. En realidad, estas casas son entidades en las que se realizará el aprendizaje del comprador por medio de las experiencias de otro agente. Una casa, al igual que una tienda, puede ser visitada por mas de un agente comprador a la vez. En las casas sucede una situación parecida al comprar un producto, es decir, en las casas conoceremos si hubo o no fraude en las ventas de esos productos, lo que llevará a una disminución del grado de confianza de ese vendedor. Caso se detecte que no hubo fraude, el grado de confianza aumentará. De esa forma podemos aprender sin necesidad de comprar, y solo por medio de las experiancia de los otros.

La simulación se inicia cuando los vendedores disponen de articulos para la venta y los compradores esten en su respectivas casas. Los compradores (conociendo ya los articulos a comprar) comienzan su busqueda por ambiente, visitando diferentes tiendas y almacenando informaciones de potenciales vendedores en una lista. Esta lista está ordenada de acuerdo a un criterio de evaluación de las caracteristicas del producto. Los compradores tienen un atributo extra, al que se llamo ***fuerza***. Esta fuerza permite al agente desplazarse por el ambiente, disminuyendo conforme el comprador se desplace y al acabarse, el agente se ve obligado a comprar, esto es, regresar a la tienda del primer vendedor de su lista y preguntar si aún posee el articulo. En caso el tenga el artículo aún, se compra y retorna a su casa. Si el vendedor ya no dispone del articulo, se va hacia el segundo de la lista y se repite el proceso. Si la lista se termino y no se compro nada, se regresa a su casa. En las figuras 3 se representa los comportamientos de los agentes.

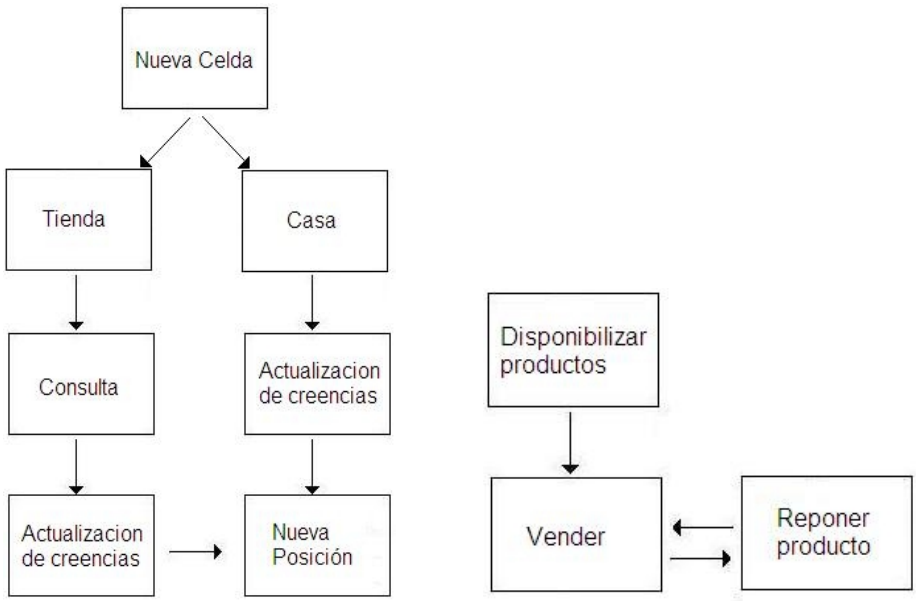


Figura 3 Comportamientos del agente comprador (derecha) y del vendedor (izquierda)

El ambiente se encargará de manejar las percepciones que cada agente recibirá conforme se desplaze por el ambiente. Por ejemplo, si el agente esta en una celda, el sabrá si es una tienda o una casa por medio de las percepciones que reciba.

4. DESCRIPCIÓN DE LOS AGENTES

Para la implementación de los agentes (comprador y vendedor) se utilizo la herramienta AgentSpeak, por ser este uno de los pocos lenguajes orientados exclusivamente al modelo BDI.

Para la implementación del ambiente se utilizo lenguaje el ELMS [5] para la implementación del ambiente, y SACI (Simple Agent Communication Infrastructure) [11] como medio por el cual se manejarán la comunicación entre los agentes (se escogio estas dos ultimas herramientas por que ya se cuenta con toda una infraestructura que facilita la programación y su unión con AgentSpeak).

4.1 Agente Vendedor

4.1.1 Estructura e Implementación

Sus componentes son:

- Articulos para vender:
  - Nombre *(n)*,que identifica al producto.
  - Valor *(v)*, cantidad de dinero que el vendedor gasto para la obtención del producto.
  - Precio *(p)*, cantidad de dinero que el vendedor solicita a los compradores a cambio del articulo. Es claro que *p* es mayor que *v*.
  - Calidad Real del producto (*cr*). Atributo numérico entre 0 y 100. A mayor valor de cr, mayor será la calidad del producto.
  - Calidad para venta *(cv)*. Atributo numérico entre 0 y 100. A mayor valor de cv, mayor será la calidad que dice tener el producto. Este número tiene que ser mayor o igual a cr.
- Dinero inicial. Dinero con el cual inicia la simulación.
- Dinero Final. Dinero con el que termina la simulación.

Este agente tiene algunos planos que son consecuencia de sus intenciones (vender productos).

Una vez iniciada la simulacion, el ambiente envia la siguiente percepción al agente:

```
+retailer(instance(INS),id(ID)): true
      <- +id(ID);
         !buyProducts(Money).
```

Esta percepción informa al agente vendedor que tiene un identificador (ID) por medio de la adición de la crencia `+id(ID)`. A su vez, se desencadena otro plan, llamado `!buyProducts(Money)`. Básicamente, lo que hace ese plan, es crear productos para la venta, a cambio de una cantidad de dinero. Esa cantidad de dinero es reducida del valor Money.

El plan buyProducts(Money), tiene la siguiente forma:

```
+! buyProducts(Money):id(ID) & retailer(instance(ID),money(M)) & .gte(M,100)
      <- //Se crean los atributos y se almacenan en las variables
         //CodArt,FalseQuality, RealQuality,Price
         new_prod(CodArt,SellQuality, RealQuality,Price);
         .subB(M,RealQuality,NewMoney);
         -money(M);
         +money(NewMoney);
         !buyProducts(NewMoney).
```

Este plan instancia las variables ID (instancia) y M (dinero) del agente vendedor, para despues hacer la comparación de M con el número 100. Esto es un criterio de parada para el bucle.

Si M es mayor que 100, se ejecuta el plan. El plan "crea" algunos articulos con sus determinadas caracteristicas para despues informar de esto al ambiente (linea new_prod(CodArt,FalseQuality, RealQuality,Price)) que se encargara de atribuir este articulo como una creencia al agente. Notese, que para generar los atributos del articulo, usamos algunas funciones que dispone el lenguaje AgentSpeak que no mostraremos aca.

Para terminar el bucle y comenzar a vender, adicionamos el siguiente plan

```
+!comprarArtigos(Money): true
      <- true.
```

Este plan, garantiza que el agente no intente comprar articulos por siempre. Una vez que se vende algun articulo, se procede a sustituirlo por otro, para ello, el siguiente plan es adicionado:

```
+retailer(instance(ID),money(M)):id(ID)
      <- //Se crean los atributos y se almacenan en las variables
         //CodArt,FalseQuality, RealQuality,Price
         new_prod(CodArt, FalseQuality, RealQuality, Price).
```

Este plan será disparado cuando se detecte alguna modificación de la cantidad de dinero M. Al igual que en el primer plan, se utiliza la acción `new_prod(CodArt, FalseQuality, RealQuality, Price)`para informar al ambiente la existencia de un nuevo producto.

4.2 Agente Comprador

4.2.1 Estructura

Los atributos del agente comprador são:

- Articulo (s) para comprar:
  - Nombre *(n),*que identifica al producto.
  - Precio *(p)*, cantidad de dinero que puede pagar como maximo por el producto.
  - Calidad Esperada *(qe)*. Atributo numérico entre 0 y 100. El producto a comprar debe tener como mínimo esa calidad.
- Grados de confianza de cada vendedor, valor numérico entre 0 y 100, que aumentará o disminuirá de acuerdo a las interelaciones con otros compradores o con el vendedor.
- Fuerza, valor numérico que permitirá el desplazamiento del agente por el ambiente.
- Lista de vendedores para el artículo actual, clasificados de acuerdo a un criterio de evaluación.
- Histórico de ventas pasadas.

Estos atributos son parte de las creencias del agente. El deseo básico del agente comprador, el adquirir todos los artículos de su lista, aunque el dinero disponible para ello sea insuficiente. La implementación de este agente, involucra aspectos como la forma de su desplazamiento, la consulta en una tienda, la visita de una casa y la compra de los artículos. Revisaremos brevemente la consulta, la visita y la compra, por ser los mas importantes.

En el caso de la consulta, el agente percibe que esta en una tienda, porque el agente informa de esto por medio de percepciones. El manejo de esta percepción esta implementado de la siguiente forma:

```
+!consulta(NUM): not(product(instance(INS),realquality(RQ))) & .gte(NUM,1) &
compra(NUM,CodArt,Price,Quality)
      <- !compararCaracteristicas(NUM,CodArt,Price,Quality);
         .subB(NUM,1,NewNum);
         !consulta(NewNum).
```

Donde NUM, representa al identificador del artículo a comprar, INS a instancia del producto a venta y RQ a calidad real. El comprador sabe que esta en una tienda porque no percibe a calidad real del producto (`not(product(instance(INS),realquality(RQ)))`), situación que si ocurre si estaría en una casa por ejemplo. Una vez que identifica que esta en una tienda, compara si los atributos del articulo que el busca coincide con alguno que ofrece a tienda. Esto es realizado por el plan `!compararCaracteristicas(NUM,CodArt,Price,Quality).` Ese plan se presenta a continuación.

```
+! compararCaracteristicas (NUM,CodArt,Price,Quality):
product(instance(INS),type(CodArt)) & product(instance(INS),price(SellPrice)) &
product(instance(INS),quality(SellQuality)) & .gte(Price,SellPrice) &
.gte(SellQuality,Quality) & product(instance(INS),retailer(Seller)) & conceptMin(CM) &
conceptSeller(Seller,CS) & .gte(CS,CM)
      <- !adicionarVendedor(Seller,NUM,SellPrice,SellQuality).
```

Lo que realiza este plan, es revisar si las caracterisiticas entre los articulos coinciden o no. Caso coincidan, se almacena la posición del vendedor en una lista de acuerdo a un criterio de evaluación.

Si el agente percibe la calidad real del producto, esto indica que se encuentra en una casa. Entonces, el actualizará sus grados de confianza con respecto a los vendedores de los artículos que están en esa casa. El plan es el siguiente:

```
+!consulta(NUM): product(instance(INS),realquality(RQ))
      <- !actualizarCrencias(INS);
         .randB(1,4,NewDirecao);
         !start(NewDirecao).
```

Segun las diferencias entre las calidades encontradas, se considera la compra como buena o como engañosa. Los planos para esas acciones son:

```
+! actualizarCrencias (NUM): product(instance(INS),id(ID)) &
product(instance(INS),realquality(QR)) & product(instance(INS),quality(Q)) &
.gte(QR,Q)
      <- ?product(instance(INS),retailer(IDR));
         ?conceptSeller(INS,Conceito);
         .addB(Conceito,X,NewConceito);
         -conceptSeller(IDR,Conceito);
         +conceptSeller(IDR,NewConceito).

+! actualizarCrencias (NUM): product(instance(INS),realquality(QR)) &
product(instance(INS),quality(Q)) & .gte(Q,QR)
      <- ?product(instance(INS),retailer(IDR));
         ?conceptSeller(INS,Conceito);
         .subB(Conceito,X,NewConceito);
         -conceptSeller(IDR,Conceito);
         +conceptSeller(IDR,NewConceito).
```

En estos planes, la variable X, indica el valor o la medida en la que variará nuestro concepto si la compra fue correcta o no.

Finalmente, cuando el vendedor detecta que el valor de fuerza es 0, tiene que comprar los artículos de su lista. Para ello tiene que reconsultar de acuerdo al orden de la lista si los vendedores aun tienen ese producto para la venta. El plan es el siguiente

```
+!reconsulta(NUM,IdVendedor): product(instance(INS),type(CodArt)) &
product(instance(INS),price(SellPrice)) & product(instance(INS),quality(SellQuality))
& .gte(Price, SellPrice) & .gte(SellQuality,Quality)  &
product(instance(INS),retailer(Seller)) & conceptMin(CM) & conceptSeller(Seller,CS) &
.gte(CS,CM)
      <-?product(instance(INS),id(IDV));
         buy(IDV,IdVendedor).
```

Las precondiciones de este plan, son parecidas al plan `compararCaracteristicas` y esto porque se tiene que consultar si el precio, calidad del articulo aun son los que el agente esperaba.

Esta parte mostro algo del codigo fuente, porque en muchas de las aplicaciones que involucran agentes BDI, se omite esta parte, lo que deja algunas dudas de como se realizan algunas operaciones basicas, sobretodo cuando se utiliza herramientas destinadas a la programación de este tipo de agentes, como es el AgentSpeak.

Informaciones sobre AgentSpeak, el lenguaje ELMS (lenguaje para la creación del ambiente) y SACI (herramienta para la tranmisión de percepciones entre ambiente y agente) pueden encontrarse en [10], [5][11] respectivamente.

5. RESULTADOS

Para efectos de la simulación se instanciaron 45 agentes compradores y 30 agentes vendedores. Estos fueron repartidos aleatoriamente por el ambiente. Para analizar el comportamiento de los agentes compradores, dividimos estos en tres tipos, a los que llamamos agentes clase A, B y C respectivamente. Ellos se diferencian, ademas de su dinero y lista de articulos, por su atributo ***fuerza***, que les permitirá desplazarse por el ambiente, visitar mas tiendas y sobretodo, visitar mas casas, lo que permitirá un mayor aprendizaje. Los agentes tipo A, tienen poca fuerza, lo que les obliga a comprar rápidamente. Los del tipo B tienen una mayor fuerza y los del tipo C mas aún (llegando casi al límite que es 100). En el caso de los agentes vendedores, dividimos a estos en dos grupos. Agentes X y agentes Y. Los del tipo X, casi no engañan (de hecho, algunos de ellos no lo hacen) y los del tipo Y, engañan casi siempre. La simulación consto de 200 ciclos, y los vendedores tienen una extensa cantidad de productos. Como métrica para el desempeño de los agentes utilizamos las siguientes fórmulas.

$$S = \frac{Ciclos\_Simulación}{Ciclos\_Compra} + Sum\left(\frac{CalidadReal}{CalidadVenta}\right)$$

Donde Ciclos_Simulación es el número de ciclos total de la simulación, ciclos_compra son los ciclos hasta que termina de comprar los artículos. En caso que la lista de vendedores acabe y el no compre la lista de artículos, el valor será 0. La sumatoria comprende los articulos comprados, donde CalidadReal es la calidad real del producto y CalidadVenta es la calidad con la que se vendio el producto.

Para los agentes vendedores, el grado de satisfacción se evaluo de la siguiente forma:

$$S1 = Dinero\_Final + Sum(Costo\_Productos)$$

Donde ***Dinero_Final*** es el dinero con el cual terminó la simulación, y ***Costo_Productos***, es el valor que el pago para disponer del producto.

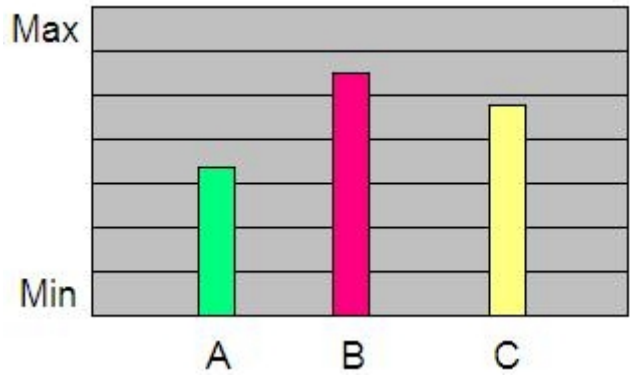


Figura 4. Indice de satisfacción de los agentes compradores

Como se puede apreciar en el diagrama, los agentes que mejor desempeño tuvieron fueron los del tipo B, es decir, aquellos con un nivel de ***fuerza*** media. Esto tiene su motivo. Los agentes del tipo A, no tuvieron el tiempo suficiente para buscar los mejores vendedores con los mejores precios a la calidad esperada, pero sobretodo, no aprendieron del ambiente, ya que en las casas que ellos visitaban, no habian artículos para la comparación de las calidades. En el caso de los agentes del tipo C, ellos tenian mucho tiempo para aprender de las compras de los otros agente, pero en muchas oportunidades, no llegaban a comprar los artículos por que otros agentes ya lo habian hecho, quedando sin vendedores, o acabando el tiempo de la simulación (200 ciclos). En el caso de los agentes del tipo B, ellos tuvieron mejor desempeño porque aprendieron de las "malas" o "buenas" experiencias de los agentes del tipo A, y cuando se decidieron a comprar, encontraron los artículos que ellos habian evaluado.

Para los agentes vendedores, se observo lo siguiente:

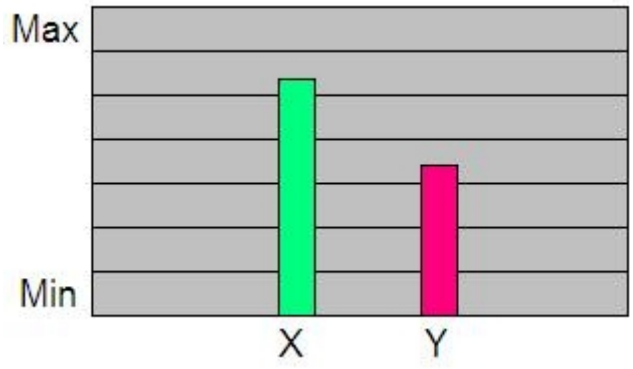


Figura 5. Indice de satisfacción de los agentes vendedores

Para los agentes vendedores, el mejor desempeño se noto con los agentes del tipo *X*, es decir aquellos que no engañan, o lo hacen poco. En cambio, los del tipo *Y*, dejaron de vender muchos de sus productos, aún cuando estuvieran en las listas de los compradores, ya que durante la simulación, por causa de los agentes del tipo A, se conocia que estos agentes engañaban con sus productos.

Se evaluo la situación de los agentes vendedores, sin la presencia de los agentes compradores del tipo A, para reconocer si ellos erán los responsables por su mala ***performance*** . En el siguiente diagrama se muestra la nueva relación entre ellos.

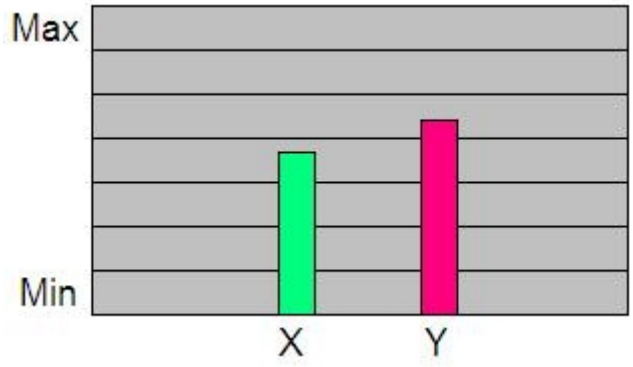


Figura 5. Indice de satisfacción de los agentes vendedores

Como se puede ver, el desempeño de los agentes del tipo *Y*, subio, pero sobre todo, el de los del tipo *X*, bajo, ya que no hubo agentes que por medio de sus compras, alimentaran al ambiente de conocimiento.

6. CONCLUSIONES

Este trabajo intento representar un modelo teórico (modelo de consumidor) en una forma práctica con la utilización de agentes inteligentes. La interacción de agentes y ambiente logro representar los aspectos mas importantes de este modelo y en base a ellos, desarrollar algunos experimentos que permitieron simular el comportamiento de los consumidores.

En lo relacionado a los resultados, se noto un mejor desempeño en los agentes compradores que pueden aprender información del ambiente, pero sin un exceso de busquedas y consultas. Sin embargo, el papel que juegan los agentes que toman desiciones apresuradas (compradores del tipo A), es vital, ya que por medio de ellos, es que el resto aprende. Desafortunadamente, ellos tienen en peor desempeño. Por el lado de los agentes vendedores, el de mejor rendimiento es aquel que no engaña, o si lo hace, lo hace poco.

Finalmente, creemos que basados en lo desarrollando hasta ahora, puede realizarse mejoras a la aplicación, lo que representaría experimetos mas complejos, como analizar la cooperación entre agente, etc.

7. TRABAJOS FUTUROS

Como se mencionó en las conclusiones, podemos hacer mucho mas complejo el comportamiento de nuestros agentes, adicionando por ejemplo intercambio de información relacionado a los articulos ofrecidos por los vendedores, lo que evitaria el tener que visitar una tienda para conocer que articulos ofrece.

Se piensa tambien, en la implementación de total de la busqueda interna por parte del consumidor (como se explico en la sección 2), posibilitando la compra de articulos sin consultar tiendas, siempre y cuando creamos que ese vendedor ofrece el artículo a condiciones adecuadas. Ello mejoraría el desempeño de los compradores, a riesgo de estar comprando un artículo a precio alto.

En cuanto a los vendedores, podríamos utilizar alguna técnica para intentar la venta de sus productos, como por ejemplo, la disminución de los precios si el producto no se vende despues de un determinado número de ciclos de simulación.

8. BIBLIOGRAFIA

[1] Bratman, M.E, Israel D.J. y Pollack M.E. Plans and resource-bounded practical reasoning. *Computational Intelligence*, 4(4):349–355, 1988.

[2] Bratman M. E. *Intentions, Plans, and Practical Reason*. Harvard University Press: Cambridge, MA, 1987.

[3] Engel, James F, Blackwell, Roger D y Miniard, Paul W. *Consumer behavior*. 8.ed. Forth Worth: The Dryden Press, 1995. 951 p. : il.

[4] Mowen, John C. y Minor, Michael. Consumer behavior. 5th ed. Upper Saddle River, N.J.: Prentice-Hall, 1998. xxi, 696 p. : il.

[5] Okuyama, Fabio Yoshimitsu. ELMS. Lenguaje para la especificación de ambientes. Tesis de maestria. 2003

[6] Rao A. y Georgeff M. Bdi agents: From theory to practice. En *Proceedings of the First International Conference on Multi-Agent Systems* - ICMAS 95, San Francisco, USA, 1995.

[7] Rao A. y Georgeff M. Modeling rational agents within a bdi-architecture. In Principles of Knowledge Representation and Reasoning: *Proceedings of the Second International Conference* - KR 91, paginas 473–484, 1991.

[8] Rao A , Lucas A. , Morley D. y Selvestrel M. Agent-oriented architecture for air combat simulation. Technical Note 42. April 1993.

[9] Rao A, Tidhar G. y Ljungberg M. Tech. Rep. 25, Australian Artificial Intelligence Institute, Melbourne, Australia, Jan 1992.

[10] Rao A. AgentSpeak(L): BDI Agents speak out in a logical computable language En: *Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World* (1996)

[11] Simão,J y Hübner J. SACI: Uma Ferramenta para Implementação e Monitoração da Comunicação entre Agentes. In: IBERAMIA/SBIA, 7, 2000, Atibaia, São Paulo. Procedings... São Carlos : ICMC/USP, 2000. p. 47-56.

Gerenciamento da Qualidade: uma nova disciplina para o RUP

Lívia N. Amorim
livia.mia@unifor.br
lnojoza@bnb.gov.br

Arnaldo D. Belchior
belchior@unifor.br

Universidade de Fortaleza - UNIFOR
Mestrado em Informática Aplicada – MIA
Av Washington Soares, 1321
CEP 60.811.341 – Fortaleza – CE – Brasil

Abstract

As the globalized society is dependent on software more and more, there is a major concern with how to get the software quality. A solution that the organizations have been fetching is the adoption of disciplined approach in order to get the guarantee of the quality of software process like ISO 9001 and CMMI. The RUP is a software engineering process that provides a disciplined approach to assigning tasks and responsibilities in the software life cycle, aiming to ensure the production of quality software. This work analyses the approach of software quality of RUP and proposes a new discipline for itself, the Quality Management, whose objective is establish the action flow that contributes effectively to the quality of software process.
Keywords: Software Engineering**,** Software Quality, Software Quality Assurance, RUP

Resumo

É crescente a dependência da sociedade globalizada em relação ao software, havendo uma maior preocupação em como atingir a qualidade de software. Uma solução que as organizações têm buscado é a adoção de uma abordagem disciplinada para garantir a qualidade de software, baseadas em processos como ISO 9001 e CMMI. O RUP é um processo de engenharia de software, que provê um enfoque disciplinado para designar tarefas e responsabilidades ao longo do ciclo de vida do software, objetivando produzir software de qualidade. Este trabalho analisa a abordagem de qualidade de software do RUP e propõe uma nova disciplina para o mesmo, o Gerenciamento da Qualidade, cujo objetivo é estabelecer o fluxo de ações que contribuem efetivamente para a qualidade do processo de software.
Palavras-Chave: Engenharia de Software, Qualidade de Software, Garantia da Qualidade de Software, RUP

1.INTRODUÇÃO

Há ainda uma grande inabilidade por parte das organizações em lidar com o complexo processo de desenvolver software. Estatísticas do chamado "Estudo do Caos" demonstram isto: 30% dos processos de software são cancelados antes do término, 189% estouram os custos, e 222% ultrapassam os prazos estabelecidos [26].

À medida que cresce a dependência da sociedade em relação ao software, a qualidade desponta como um fator essencial no desenvolvimento de produtos de software, com uma maior disposição para esse tipo de investimento [7].

As freqüentes discussões sobre a importância da qualidade para o processo de desenvolvimento de software, e para o produto dele resultante, levaram a indústria a se aliar à academia para o desenvolvimento de padrões e modelos, que pudessem garantir a qualidade de software desejada para as suas necessidades [19, 20, 22]. Modelos têm proposto a existência da função de Garantia de Qualidade de Software ou SQA (Software Quality Assurance) [4, 5, 19, 20, 23].

Este trabalho analisa a abordagem de qualidade de software do RUP, e propõe uma nova disciplina (fluxo de trabalho) para o mesmo. A disciplina denominada Gerenciamento da Qualidade objetiva estabelecer o fluxo de ações que contribuam efetivamente para a qualidade do processo e do produto de software. Essa disciplina está baseada no CMMI (Capability Maturity Model Integration) [4; 5], SWEBOK (Software Engineering Book of Knowledge) [1], e em alguns padrões para garantia de qualidade, como o IEEE Std 1028-1997 [10] e IEEE Std 1061-1998 [11].

Este trabalho está organizado como se segue. A seção 2 discorre sobre a função de Garantia da Qualidade de Software. A seção 3 apresenta a visão de qualidade de software do RUP. A seção 4 propõe a nova disciplina para o RUP: o Gerenciamento da Qualidade. A seção 5 mostra o estágio da implantação da proposta. A seção 6 apresenta as conclusões finais.

2. A GARANTIA DA QUALIDADE DE SOFTWARE

Garantir a qualidade envolve não apenas a definição do processo de software da organização, mas também a implementação de um processo de garantia da qualidade que seja adequado às suas necessidades, com a realização da garantia da qualidade do processo e do produto [22]. A qualidade do produto de software está estreitamente relacionada com a qualidade do processo que o desenvolveu [3; 25].

A estruturação da função de garantia da qualidade de software ou SQA em uma organização permite acompanhar se o processo de software definido está sendo realmente utilizado e quão aderentes estão as práticas dos projetos ao processo estabelecido. A forma de como essa função é definida depende dos objetivos da organização, assim como do conceito de qualidade de software por ela adotado [17].

Dependendo da política estabelecida pela organização, os responsáveis pela execução de SQA podem trabalhar de forma exclusiva ou em dedicação de tempo parcial [23], tendo esta atribuição institucionalizada na organização por meio da criação do grupo de SQA [19, 20, 23].

A discussão em torno do caráter de disciplina ou da definição da função de SQA na organização é tratada por Baker [2], que chega à conclusão de que é difícil caracterizar SQA como uma disciplina (do ponto de vista da formalização do processo de SQA), devido às diferentes práticas das instituições em relação à SQA, e que nem sempre SQA é uma organização (grupo).

Independente da formalização da função de SQA, a qualidade de software é responsabilidade de todos os envolvidos direta e indiretamente com o processo de software da organização, e possui impacto direto na qualidade do produto gerado. Os gerentes devem estar cônscios da importância da qualidade para se comprometerem com suas atividades e seus resultados. Os atores do processo, como gerentes de projeto, analistas, projetistas e desenvolvedores, devem contribuir para a qualidade em cada ação realizada. Também é importante o papel do cliente e/ou usuário, definindo os requisitos de qualidade para o software, e validando suas especificações em relação ao produto, o que ocorre com mais freqüência em processos de ciclo de vida iterativo [15, 27].

A garantia da qualidade do processo de software visa prover evidências sobre a capacidade do processo em produzir determinado produto, identificando falhas nesse processo e buscando resolvê-las antes que impactem no produto. A garantia da qualidade do produto de software, por sua vez, visa garantir que o software produzido esteja em conformidade com requisitos funcionais e de desempenho especificados; atenda aos padrões de desenvolvimento documentados; seja o mais isento possível de erros; e atenda às características implícitas esperadas pelo usuário.

As atividades de qualidade do processo de software compreendem a definição do processo, a verificação da utilização do processo definido, e a preocupação com a melhoria do processo utilizado. Primeiramente, o processo da organização deve ser definido, atendendo os objetivos da organização. No CMMI, por exemplo, este trabalho é conduzido pela equipe de processo de engenharia de software (SEPG), com a participação da equipe de SQA [4, 5, 19, 20]. Enquanto o SEPG é responsável pelo processo, o SQA verifica a aderência das práticas de engenharia de software dos projetos ao processo definido.

São utilizadas técnicas de SQA para verificar se o processo definido está sendo seguido, para encontrar defeitos diretamente no produto, ou para indicar a necessidade de um exame mais detalhado no mesmo. As principais técnicas utilizadas são as Revisões de Software, que são classificadas como Auditorias, Inspeções, Walkthroughs, Revisões Técnicas e Revisões Gerenciais [10]. As Revisões de Software avaliam o processo de desenvolvimento, seu gerenciamento, e o produto de software [1, 10]. Além das Revisões, são também realizados Testes, os quais constituem um dos elementos críticos da garantia de qualidade de software e representam a última revisão das especificações de projeto e código do produto de software.

Quando o foco é a melhoria do processo estabelecido, são utilizados as Avaliações (Assessment), que examinam um processo para determinar a sua capacidade, por meio da comparação desse processo em relação a um padrão de melhores práticas [8, 19, 20].

Alguns modelos e normas internacionais tratam a qualidade do processo de software, e trazem, em seu escopo, a figura da função de qualidade. O CMMI [4, 5] serve para determinar a capacidade e maturidade de uma organização para a produção de software. A norma ISO 9001 trata da definição, implementação e manutenção do Sistema de Gestão da Qualidade para uma organização [13]. A ISO/IEC 15504 constitui-se de um framework para modelos de avaliação de processos de software, podendo ser utilizado também em estratégias para melhoria de processos [14].

A literatura em qualidade de software relata ainda algumas propostas para formalização do processo de qualidade de software. Marczak et al. [17] propõem um modelo de organização da função de SQA, considerando aspectos práticos da implantação da garantia de qualidade em uma organização de desenvolvimento de software, no contexto de implantação do CMM-SW [17], baseado na área chave de processo SQA desse modelo. Unhelkar [27] estabelece um processo de qualidade de software para organizações que trabalhem com projetos baseados na UML (Unified Modeling Language) [27]. Segundo Unhelkar [27], a arquitetura do processo de qualidade de software é constituída por três componentes: a Gestão de Qualidade, a Garantia da Qualidade, o Controle de Qualidade.

No mesmo enfoque de tratar a qualidade do processo de desenvolvimento, para que se gerem produtos de qualidade, surgiu a abordagem do framework RUP, que visa assegurar o desenvolvimento de produtos de software de qualidade, que satisfaçam as necessidades de seus usuários finais, dentro de cronogramas e orçamentos previsíveis. Para alcançar este objetivo, o RUP diz utilizar-se de melhores práticas em desenvolvimento de software, de forma a ser adaptado para uma grande variedade de projetos e organizações. São elas [15]: desenvolver software iterativamente; gerenciar requisitos; usar arquitetura baseada em componentes; modelar software visualmente; verificar a qualidade de software; controlar mudanças no software [15, 21].

3. A VISÃO DE QUALIDADE DE SOFTWARE NO RUP

O RUP é uma abordagem de desenvolvimento de software iterativa. É um processo de engenharia de software, que estabelece o que deve ser feito, como deve ser feito, quando e por quem, abrangendo o ciclo de vida de software de um projeto. Provê um framework de processos de software, podendo ser utilizado por uma organização em seus projetos de desenvolvimento de software, por meio da adaptação desses processos à sua realidade [15].

O RUP compõe-se de duas dimensões ou estruturas: uma estática e outra dinâmica, que podem ser visualizadas na Figura 1.

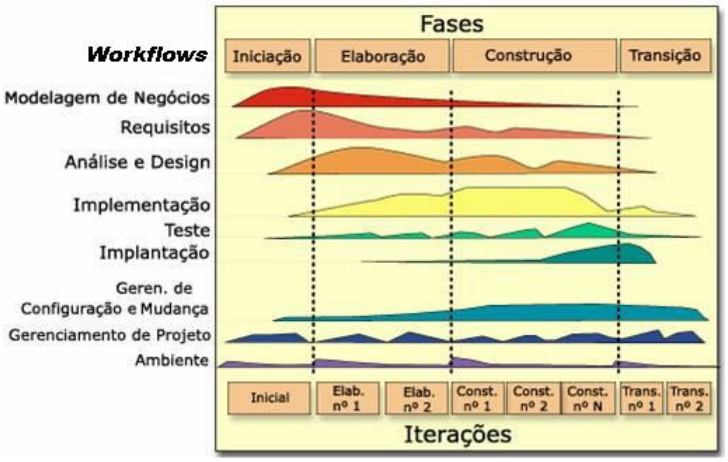


Figura 1: Dimensões do RUP

A dimensão estática representa a estrutura estática do processo, descrevendo como os elementos do processo são agrupados logicamente em disciplinas (workflows). Disciplinas são agrupamentos lógicos de papéis, atividades, artefatos e outros guias para a descrição de um processo, e são representadas por um fluxo de trabalho.

Tabela 1: Elementos do processo do RUP

Elemento	Conceito
Papel	Definição do comportamento e das responsabilidades de um individuo ou grupo de indivíduos trabalhando juntos em equipe
Atividade	É uma unidade de trabalho, executada por um papel com um propósito definido, geralmente criando ou atualizando um artefato. Pode ser dividida em passos (*steps*).
Artefato	Um pedaço de informação que é produzido, modificado ou criado por um processo. São usados como entrada na execução de atividades e como produto das atividades executadas.
Conceito	Introduzem definições e princípios chave.
Guia	Prover regras, recomendações e heurísticas que suportam atividades, passos e artefatos.

A dimensão dinâmica do RUP é representada pelo tempo e expressa o processo por meio de ciclos, decompostos em fases, que são divididas em iterações com marcos de conclusão. Essas fases são: Concepção, Elaboração, Construção, e Transição [18]. Cada fase possui objetivos de negócio e contém iterações que focam na produção de artefatos técnicos para atingir aqueles objetivos. As descrições de cada fase são sumarizadas na Tabela 2.
No que diz respeito à qualidade de software, a visão apresentada pelo RUP vem evoluindo em relação à qualidade do processo e à qualidade do produto. Nele, a qualidade é responsabilidade de todos os envolvidos no processo, mas o papel do gerente de projeto possui destaque como responsável pela gestão da qualidade [15, 21].
A qualidade do processo no RUP é implementada ao longo das disciplinas, fases e iterações do ciclo de vida do projeto, estando especialmente ligada à disciplina de Gerenciamento do Projeto. O gerente de projeto é o papel responsável por estabelecer as atividades relacionadas à qualidade de software no projeto. Há revisões técnicas e revisões gerenciais. As primeiras são realizadas sobre os artefatos, como requisitos, modelos de casos de uso e código. As outras são conduzidas pelo gerente de projetos, para avaliar o status das atividades de projeto pelo menos uma vez a cada iteração, e pelo PRA (Project Review Authority) para revisar a conformidade do projeto com obrigações contratuais e com padrões organizacionais [21].

Tabela 2 - Fases do RUP

Fase	Descrição
Concepção	Estabelece uma boa compreensão do sistema a ser construído, por meio de requisitos de alto nível e do estabelecimento do escopo do projeto. Obter acordo com os envolvidos (*stakeholders*), para prosseguir ou não com o projeto, também caracteriza essa fase. Também é preparado o ambiente para o projeto, estabelecendo o caso de desenvolvimento.
Elaboração	O propósito desta fase é compreender como construir, analisando o domínio do problema, estabelecendo uma arquitetura básica, desenvolvendo um plano de projeto, e eliminando os maiores riscos do projeto.
Construção	Os componentes da solução são desenvolvidos e integrados em um produto. Custos, cronograma e qualidade são otimizados pelo gerenciamento dos recursos e o controle das operações. Os componentes são desenvolvidos e testados baseado em critérios de avaliação definidos. *Releases* do produto também são avaliadas.
Transição	O principal objetivo desta fase é construir a versão final do produto e disponibilizá-la para os usuários finais. São desenvolvidos os materiais de suporte à instalação e uso pelo usuário final.

Também relacionado à qualidade do processo, há o papel do SEPA (Software Engineering Process Authority), que auxilia a organização na identificação dos processos dos projetos, avaliando-os continuamente com o intuito de melhorá-los. Para estabelecer o processo a ser utilizado no projeto, o RUP é adaptado para o projeto pelo engenheiro de processos, pelo gerente do projeto ou até mesmo por um mentor, que é alguém com experiência em implementações do Processo Unificado [21].

No entanto, a ênfase de qualidade no RUP está voltada para o produto. O RUP aborda a qualidade do produto como sendo algo verificável por meio dos testes realizados nos releases ao longo do ciclo de vida do projeto. Como a abordagem é iterativa e os testes são realizados em cada fase, a qualidade do produto é verificada continuamente, sendo possível identificar defeitos mais cedo [15]. Doria [6] afirma que existem amplas referências na literatura de que, dependendo do caso, as inspeções podem ser mais efetivas que os testes e a um custo bem menor.
O modelo FURPS+ [9], aplicado para a caracterização de requisitos, é usado no RUP também para avaliar a qualidade do produto. O RUP faz uma distinção entre os dois tipos de produtos: o software executável, versão final disponibilizada ao cliente, e os produtos de software, que resultam do processo de desenvolvimento, como os planos, as especificações e outros artefatos (byproducts), que estão associados ao processo [15].
Apesar de considerar a importância da qualidade do processo e das atividades de revisão, Kroll & Kruchten [15] afirmam que para o RUP, "at the end of the day, what really counts is how good your code is, not how good your byproducts of software development are".
Os principais artefatos relacionados à qualidade de software no RUP [21] são:

- Plano de Garantia de Qualidade: elaborado por projeto e estabelecido pelo gerente do projeto, para registrar as ações de qualidade previstas e realizadas, especificando como assegurar a qualidade do produto, dos artefatos e do processo.
- Caso de Desenvolvimento: elaborado por projeto, contém a definição do processo a ser seguido pelo projeto, a partir das adaptações do RUP.
- Registro de Revisão: captura os resultados da atividade de revisão no qual um ou mais artefatos dos projetos são revisados. Segundo o RUP [21], "é mais importante realizar a revisão do que documentá-la", sugerindo que um e-mail sirva como registro da revisão.

Alguns pesquisadores apontam deficiências no RUP na abordagem de qualidade de software. Manzoni & Price [16] analisam que o RUP não estabelece como tratar os problemas identificados nas revisões e nem como os resultados das revisões são comunicados aos demais envolvidos com o projeto de software e aos envolvidos com o processo de software.
Para Smith [24], que analisou as características do RUP em relação ao padrão IEEE 1074 [12], o RUP possui várias carências abrangendo: a formalização das avaliações de qualidade, especialmente em termos dos tipos de revisões realizadas; a ausência de aprovação de desvios identificados em relação ao processo padrão (framework do RUP); e ao uso de avaliações com foco na melhoria do processo de software utilizado. Esta análise sugere que os aspectos de qualidade abordados na disciplina de testes, que são bem cobertos, sejam aliados à abordagem das avaliações de revisão ao longo do processo, que são tratadas informalmente no RUP, e constituam uma nova disciplina de avaliação.
Neste contexto, este trabalho propõe uma nova disciplina para o RUP: o Gerenciamento da Qualidade, cujo objetivo é estabelecer o fluxo de ações que conduzam à qualidade do processo de software, suprindo algumas das deficiências encontradas no RUP em relação à garantia da qualidade de software.

4. GERENCIAMENTO DA QUALIDADE: DISCIPLINA DE QUALIDADE DE SOFTWARE NO RUP

No RUP, a qualidade do produto é tratada de forma abrangente pela disciplina de Testes, e é não dada a relevância necessária ao tratamento dado à qualidade do processo de software. Neste contexto, é proposta a disciplina de Gerenciamento da Qualidade para o RUP, que visa estabelecer o fluxo de ações que contribuam efetivamente para a qualidade de software, por meio do planejamento, da garantia, e da monitoração da qualidade ao longo do processo de desenvolvimento.
A disciplina de Gerenciamento da Qualidade (GQ) está estruturada de modo a atender os seguintes objetivos:

- Prover visibilidade apropriada para a qualidade do processo de desenvolvimento no RUP, destacando a importância da mesma em cada iteração.
- Sistematizar e formalizar ações de qualidade do processo de software, executadas mediante o planejamento da qualidade realizado em paralelo com o planejamento do projeto. São estabelecidas as principais técnicas de SQA, conforme o padrão IEEE Std 1028-1997, que são utilizadas durante as iterações para verificar a qualidade do processo de software, suprindo a deficiência relatada por Smith [24].
- Tratamento sistemático das não conformidades, facilitando sua identificação, seu relato aos envolvidos, e a identificação de ações corretivas e preventivas (quando possível). Isto não abordado no RUP, segundo Manzoni & Price [17].
- Possibilitar que as ações de qualidade estejam alinhadas aos objetivos de qualidade especificados para a organização. A execução das atividades de SQA é realizada de acordo com o planejamento da qualidade de software para o projeto e para a iteração. O planejamento da qualidade, por sua vez, considera os objetivos

de qualidade especificados para a organização, para o projeto e ainda para a iteração. Isso possibilita que as ações em prol da qualidade sejam realizadas de forma contínua, estritamente alinhadas aos objetivos estabelecidos, e tendo seu alcance monitorado.

- Permitir a utilização de métricas de processo relacionadas aos objetivos de qualidade para a iteração, para o projeto e para a organização.
- Contribuir para as ações de melhoria do processo de software no RUP. A disciplina proposta estabelece o registro das lições aprendidas ao longo do processo. Esta atividade, aliada ao uso sistemático de métricas, facilita a identificação da aderência de boas práticas de engenharia de software do projeto em relação ao processo configurado a partir do framework do RUP, municiando a organização e, especialmente, o SEPA (Software Engineering Process Authority) e o Engenheiro de Processos do RUP com as informações necessárias para propor a instância de processo mais adequada às necessidades do projeto.

Para que a nova disciplina proposta para o RUP, Gerenciamento da Qualidade, exerça efetivamente seu papel nos projetos de software, urge que seja estabelecido um conjunto de atividades de "gerenciamento da qualidade" no âmbito organizacional. Essas atividades devem ser realizadas necessariamente antes da instanciação da disciplina ao nível de projeto. As principais atividades que devem ser realizadas no âmbito institucional são:

- definição dos objetivos de qualidade e de políticas de qualidade para a organização;
- estabelecimento de procedimentos, padrões e métricas institucionais de qualidade;
- especificação de papéis e da função da qualidade na organização;
- formatação do perfil para os profissionais envolvidos no processo de software;
- realização de treinamentos para os envolvidos no processo, para a disseminação dos conceitos de qualidade para a organização;
- o estabelecimento de controles sobre o custo das atividades de qualidade nos projetos.

Para a disciplina Gerenciamento da Qualidade são propostos dois novos papéis para o RUP: o Gerente de qualidade, e o Engenheiro de qualidade.

O Gerente de Qualidade deve administrar a função de qualidade de software da organização, mantendo os objetivos organizacionais da qualidade alinhados aos objetivos de negócios organizacionais e avaliando se os projetos estão alcançando os objetivos de qualidade. Deve possuir experiência com gestão de processos e de pessoas, conhecimento da organização e conhecimento sobre modelos e padrões de qualidade de software.

O Engenheiro de Qualidade deve ter forte embasamento em engenharia de software, mormente em modelos, padrões e técnicas de SQA, e como estes elementos estão relacionados com o processo de desenvolvimento de software.

Ambos, Gerente de Qualidade e Engenheiro de Qualidade, devem estar cientes do papel da função de qualidade de software no contexto organizacional e dos projetos, podendo agir como consultores de engenharia de software para projetos da organização, e provendo orientação a respeito dos padrões organizacionais e de qualidade.

O fluxo da disciplina Gerenciamento da Qualidade é apresentado na Figura 2, e possui as macro-atividades a seguir: (i) Planejar Qualidade do Projeto; (ii) Planejar Qualidade da Iteração; (iii) Garantir Qualidade; (iv) Monitorar Qualidade; e (v) Registrar Lições Aprendidas. O conjunto de atividades referentes aos itens (ii), (iii) e (iv) deve ocorre para cada iteração do RUP ao longo do processo de desenvolvimento. A seguir, são explanados os detalhes de cada parte do fluxo da disciplina.

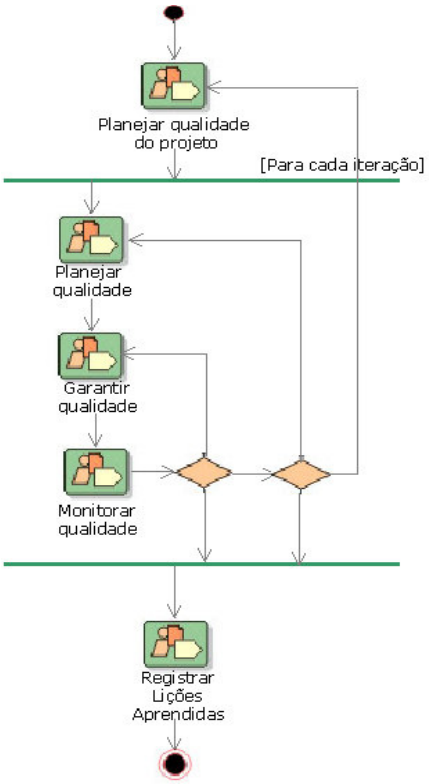


Figura 2 - Fluxo da disciplina Gerenciamento da Qualidade

A Figura 3 detalha as atividades do fluxo Planejar Qualidade do Projeto, que devem ser executadas em paralelo com as atividades de planejamento do próprio projeto. Nesse fluxo, são definidos os objetivos de qualidade de software para o projeto, com base nos requisitos do projeto, plano de desenvolvimento e padrões de qualidade da organização. São estabelecidas as métricas de qualidade associadas a cada objetivo de qualidade especificado, para que seja possível monitorar se as metas foram atingidas.

O Engenheiro de qualidade planeja, a partir dos objetivos de qualidade, que atividades de qualidade serão executadas ao longo do projeto e constrói o cronograma dessas atividades. Estas informações devem ser registradas no Plano de Garantia de Qualidade de Software do projeto. Ele também pode orientar integrantes do projeto a respeito dos padrões de qualidade adotados e das práticas de engenharia de software aplicáveis.

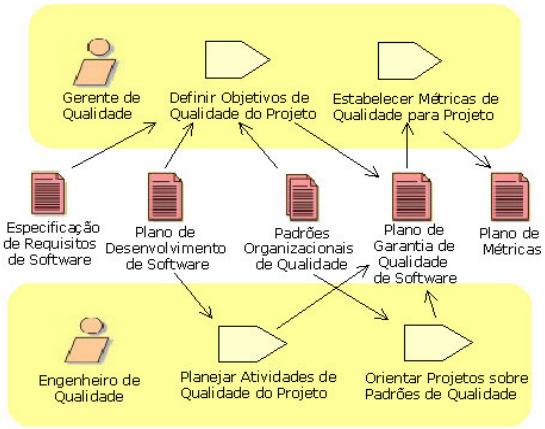


Figura 3 - Detalhe do fluxo: Planejar Qualidade do Projeto

A Figura 4 detalha o fluxo Planejar Qualidade na Iteração. Ao se iniciar uma iteração, devem ser identificados, dentre os objetivos de qualidade do projeto levantados na macro-atividade anterior, quais aqueles que serão abordados durante a iteração. Em seguida, deve ser elencadas as atividades de qualidade de software a serem realizadas durante a iteração, atualizando-se o Plano de Garantia de Qualidade.
A partir da segunda iteração, com os dados provenientes da macro-atividade de Monitorar Qualidade, podem ser especificadas as ações preventivas para as não conformidades (reconhecimento da falta de aderência de determinado item sob garantia de qualidade em relação aos padrões de qualidade) identificadas na iteração anterior, minimizando assim a ocorrência de não conformidades.

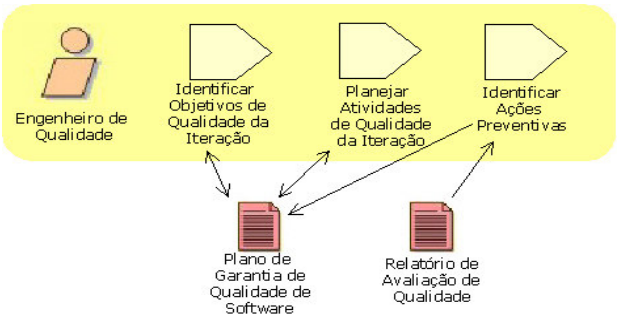


Figura 4 - Detalhe do fluxo: Planejar Qualidade na Iteração

A Figura 5 apresenta o detalhamento do fluxo Garantir Qualidade na iteração, que é realizada através de revisões de atividades do processo, por meio da avaliação de artefatos. As revisões e avaliações são conduzidas a partir do padrão IEEE 1028 [10]. O Relatório de avaliação da qualidade do projeto registra as informações sobre as revisões e avaliações realizadas, os itens avaliados, as não conformidades identificadas, o grau de severidade a elas associadas, o responsável pela resolução da não conformidade, assim como os prazos para que isso seja executado. Esse relatório formaliza o Registro de Revisão do RUP. As não conformidades identificadas e reportadas devem ser acompanhadas pelo Engenheiro de Qualidade até seu desfecho. Ainda são coletadas métricas de qualidade do projeto, e são fornecidas orientações (quando necessário) sobre padrões e práticas de engenharia de software.

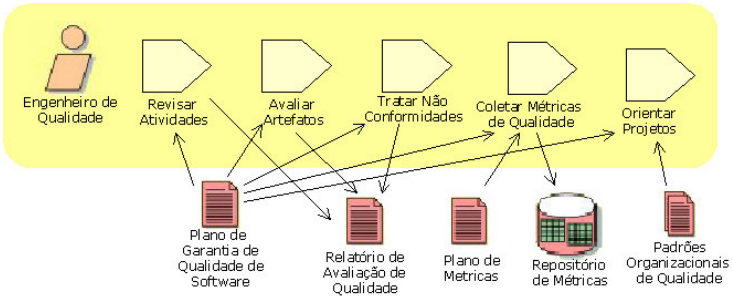


Figura 5 - Detalhe do fluxo: Garantir Qualidade

A Figura 6 apresenta o detalhamento do fluxo Monitorar Qualidade na iteração. O Engenheiro de qualidade analisa dados sobre as atividades de qualidade realizadas (revisões de atividades, avaliações de artefatos e tratamento de não conformidades) e sobre as métricas coletadas, com o objetivo de adequar ou melhorar o processo de qualidade, e permitir que ações preventivas por ser empreendidas.
O Gerente de qualidade avalia o alcance dos objetivos de qualidade da iteração e do projeto, e o desempenho da equipe de qualidade alocada ao projeto. O Relatório de garantia de qualidade é o artefato por meio do qual são reportadas as atividades realizadas pela equipe de qualidade para todos os envolvidos com o processo, dando visibilidade ao trabalho da equipe de qualidade e à importância da função de qualidade de software para a organização e para os projetos.

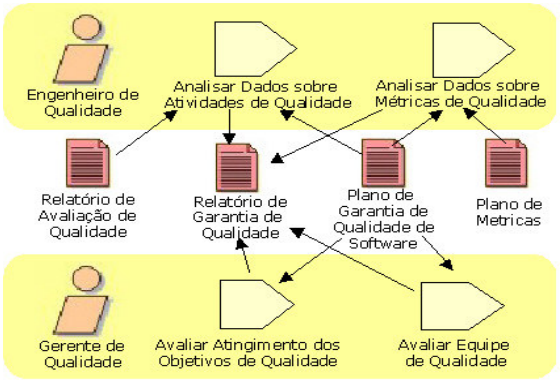


Figura 6 - Detalhe do fluxo: Monitorar Qualidade

Ao final do projeto, são registradas as lições aprendidas (Figura 7) do processo de qualidade do projeto. Essas lições podem:

- auxiliar o Engenheiro de processo, na disciplina de Ambiente, a selecionar a instância do framework do RUP, que constituirá o processo mais adequado para projetos similares, atualizando o artefato Caso de Desenvolvimento;
- identificar tipos de não conformidades mais comuns e como elas podem ser prevenidas e corrigidas;
- possibilitar avaliar como determinados tipos de não conformidades podem ser resolvidos e em que espaço de tempo;
- ajudar a quantificar a equipe de qualidade adequada por tipo de projeto, dentre outras informações válidas.

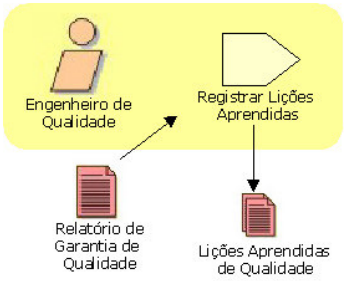


Figura 7 - Detalhe do fluxo: Registrar Lições Aprendidas

Cada uma das atividades constantes da disciplina de Gerenciamento da Qualidade é detalhada em termos de propósitos, passos, artefatos de entrada e de saída, freqüência de execução da atividade, e atores envolvidos. No Quadro 1, é descrita a atividade Definir objetivos de qualidade do projeto, do fluxo Planejar Qualidade do Projeto (Figura 3).

<table>
<tr><th colspan="2">Atividade: Definir objetivos de qualidade do projeto</th></tr>
<tr><td colspan="2">Propósito: estabelece os objetivos de qualidade de software para o projeto, observando as definições organizacionais para a qualidade e as características específicas do projeto (requisitos legais, padrões do cliente do projeto etc.)</td></tr>
<tr><td colspan="2">Passos:
- verificar os padrões organizacionais de qualidade (Políticas, padrões, procedimentos etc);
- identificar requisitos de qualidade do projeto (requisitos do cliente, legislação etc.);
- definir os objetivos de qualidade de software do projeto, orientando os projetos quanto às suas necessidades de qualidade de software.</td></tr>
<tr><td>Artefatos de Entrada:
- Padrões organizacionais de qualidade;
- Plano de Desenvolvimento de Software;
- Especificação de Requisitos de Software; e FURPS+.</td><td>Artefatos de Saída:
- Plano de Garantia de Qualidade de Software.</td></tr>
<tr><td colspan="2">Freqüência: a cada reunião de planejamento do projeto.</td></tr>
<tr><td colspan="2">Ator: Gerente de Qualidade</td></tr>
</table>

Quadro 1 - Definir objetivos de qualidade do projeto

5. Implementação da Disciplina Gerenciamento da Qualidade

A idéia da formalização da função de SQA no RUP nasceu a partir da experiência dos autores com a implementação do RUP em uma organização que, simultaneamente à implantação do RUP, também estava buscando implantar a função de SQA incorporada à Gestão da Qualidade. Essa organização atualmente já é certificada CMM nível 2.

Uma segunda organização, na qual está sendo implementada a nova disciplina proposta (Gerenciamento de Qualidade) adotou o framework do RUP na definição de seus processos de software e está buscando a certificação CMM nível 2.

A implementação da função de SQA nesta organização está utilizando a disciplina proposta como processo de qualidade de software. O processo de implementação está em andamento, no estágio de realização das atividades em âmbito organizacional, tendo já sido definidos os objetivos da qualidade, Procedimentos de Qualidade e a Política de Qualidade da organização. Estão sendo definidos os padrões organizacionais de qualidade.

Foram especificados os papéis necessários ao processo de desenvolvimento e estão sendo realizados treinamentos para disseminação da cultura e conceitos de qualidade na organização.

Foi criada uma unidade organizacional para englobar a função de SQA, que é independente da estrutura que contém os projetos que serão avaliados em relação à qualidade, garantindo a independência de atuação do Engenheiro de qualidade e Gerente de qualidade.

Com a conclusão dos treinamentos e da revisão dos padrões, as atividades da disciplina serão instanciadas em alguns projetos selecionados como piloto. Já é possível observar a importância da formalização da disciplina de qualidade de software no dia-a-dia da organização, por meio da identificação de não conformidades e dos pontos de controle que precisam ser melhorados no processo de software utilizado.

6. Conclusão

A proposta de inclusão no RUP da disciplina de Gerenciamento da Qualidade traz como principais contribuições:

- Formalização de um processo de qualidade de software, dentro de um contexto de ciclo de vida iterativo / incremental, para projetos que sigam o processo de desenvolvimento baseado no framework do RUP;
- Fortalecimento da perspectiva de qualidade do processo de software do RUP, formalizando ações e alinhando-as a um processo de gestão da qualidade;
- Expansão da visão de qualidade do produto, que era mais focada no software executável, também para os demais produtos de software (byproducts) produzidos ao longo do processo de desenvolvimento.
- Estabelecimento do uso e da análise de métricas mais alinhadas aos objetivos da qualidade.
- Enfatiza ações que trabalham a melhoria do processo de software para o projeto, e não apenas as mudanças de configurações de processo, possibilitando o uso do processo melhorado em projetos similares.

- Pode auxiliar a organização na estruturação de seus processos rumo à certificação CMMI nível 2, já que a proposta considera várias práticas exigidas por este modelo.

Como trabalhos futuros, pode-se elencar a avaliação das características de qualidade de software do RUP em relação a modelos de Qualidade de Software, como o CMMI e a ISO 15504, a fim de possibilitar às organizações que utilizam o RUP usufruírem uma implementação robusta de SQA e da Gestão da Qualidade.

Referências Bibliográficas

[1] A Guide for Software Engineering Body of Knowledge, IEEE – Trial (Version 0.95) – May, 2001.

[2] Baker, Emanuel R., Which Way, SQA? IEEE Software January/February 2001.

[3] Conradi, R. & Fuggetta, A., Improving Software Process Improvement. IEEE Software July/August 2002.

[4] CMMI Product Team. CMMI for Systems Engineering/Software Engineering, Version 1.1 Staged Representation (CMU/SEI-2002-TR-029, ESC-TR-2002-029). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, August, 2002.

[5] CMMI Product Team. CMMI for Systems Engineering/Software Engineering, Version 1.1 Continuous Representation (CMU/SEI-2002-TR-028, ESC-TR-2002-028). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, August, 2002.

[6] Dória, Emerson, Replicação de estudos empíricos em engenharia de software. 1v. 154p. Dissertação de Mestrado. Universidade de São Paulo/São Carlos, 2001.

[7] Duarte, Cristina & Falbo, R.A., Uma ontologia de qualidade de software. Workshop de Qualidade de Software, João Pessoa, p. 275-285, Outubro de 2000.

[8] Dunaway, D. & Masters, S., CMM-Based Appraisal for Internal Process Improvement (CBA IPI): Method Description (CMU/SEI-96-TR-007). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, April 1996.

[9] Grady, Robert, Practical Software Metrics for Project Management and Process Improvement. Prentice-Hall, 1992.

[10] IEEE Std 1028–1997, IEEE Standard for Software Revisions.

[11] IEEE Std 1061–1998, IEEE Standard for a Software Quality Metrics Methodology.

[12] IEEE Std 1074-1997, IEEE Standard for Developing Software Life Cycle Process.

[13] ISO 9001, Sistemas de Gestão da Qualidade – Requisitos. 2000.

[14] ISO/IEC TR 15504, Software Process Assessment, 2003.

[15] Kroll, Per & Kruchten, Philippe, The Rational Unified Process made easy: a practitioner's guide to the RUP. Pearson Education, 2003.

[16] Manzoni, Lisandra V. & Price, Roberto T., Identifying Extensions required by RUP (Rational Unified Process) to comply with CMM (Capability Maturity Model) Levels 2 and 3. IEEE Transactions on Software Engineering, Vol. 29. No. 2, Fevereiro de 2003.

[17] Marczak, S., Sá, L. Ceccato, I., Audy, J. & Antunes, D., Uma proposta de organização e funcionamento da função de garantia de qualidade de software em um contexto de implantação do SW-CMM. II Simpósio Brasileiro de Qualidade de Software. Fortaleza, Brasil. 2003.

[18] Mohaghegi, Parastoo, Software Engineering Processes RUP and XP. Lecture Notes to IKT-2340 "Open Systems Seminar" at Hogskolen Agder, 2002. Disponível em http://fag.grm.hia.no/ikt2340/year2002/themes/process/notes/2-RUP-XP.pdf. Acessado em janeiro de 2004.

[19] Paulk, M. C.; Curtis, B.; Chrissis, M. B.; & Weber, C. V., Capability Maturity Model for Software. Version 1.1 (CMU/SEI-93-TR-024, ADA 263403). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1993.

[20] Paulk, M. C. et. Al., Key Practices of the Capability Maturity Model, Version 1.1. 1993.

[21] Rational Software Corporation, Rational Unified Process, Version 2003.06.00.65, CD-ROM, Rational Software, Cupertino, California, 2003.

[22] Rocha, A. R. C. et al., Qualidade de software: Teoria e prática. Prentice Hall, 2001.

[23] Schulmeyer, Gordon & McManus, James, Handbook of Software Quality Assurance. New Jersey: Prentice Hall, 1999.

[24] Smith, John., Inconsistencies found in RUP Test Drive with IEEE 1074. IBM by Chuck Walrad of Davenport Consulting, Inc., 2003. Disponível em: http://www.p1074-workgroup.org/3M_Compatibility_Team/references/Annex%20A%20table-RUP.doc. Acessado em dezembro de 2003.

[25] Sommerville, Ian, Software Engineering. 6ª ed. Addison-Wesley Pub Co, 2000.

[26] The Chaos Study, The Standish Group International, Inc., Dennis, MA. 1994.

[27] Unhelkar, Bhuvan, Process Quality Assurance for UML-based projects. In: The Addison-Wesley.

Da especificação à verificação de agentes móveis - Um ambiente gráfico

André G. Andrade*
Universidade de São Paulo, IME,
São Paulo, Brasil
oberon@ime.usp.br

and

Ana C. V. de Melo
Universidade de São Paulo, IME,
São Paulo, Brasil
acvm@ime.usp.br

and

Marcelo M. Amorim
Universidade de São Paulo, IME,
São Paulo, Brasil
mamorim@ime.usp.br

Abstract

Neste trabalho, apresentamos um ambiente de especificação e verificação para π-calculus. O ambiente proposto está inicialmente constituído de dois módulos: (i) especificação e representação gráfica; (ii) simulação e verificação de equivalências. O primeiro módulo permite a especificação de processos em π-calculus com recursos de visualização e representação gráfica da especificação. O seguinte torna possível a simulação e verificação de bi-simulações entre processos descritos em π-calculus utilizando uma nova abordagem através de técnicas de normalização e bi-simulação *up-to* em verificações baseadas em autômatos.

Keywords: Verificação Formal, Agentes Móveis, Pi-Calculus, Especificação Formal

1 Introdução

Fatores como o crescimento da comunicação celular, redes locais sem fio, e serviços via satélite, nos quais as informações e recursos podem ser acessados e utilizados em qualquer lugar e a qualquer momento, têm gerado grande impacto na área de computação concorrente. Programas executados em sistemas paralelos ou distribuídos são claramente mais difíceis de ter seus comportamentos previstos que sistemas seqüenciais, devido aos mecanismos de composição inerentes aos mesmos. Agentes móveis e colaboradores, além de possuírem as dificuldades apresentadas por sistemas concorrentes, carregam ainda problemas relacionados ao fato deles próprios, e o ambiente no qual estão inseridos, poderem ser reconfigurados dinamicamente (semântica de processos de alta-ordem [21, 18]).

O desenvolvimento formal (ou semi–formal) de agentes móveis requer um fundamento matemático que dê suporte à especificação e verificação tanto dos agentes individuais quanto do sistema como um todo. O π-calculus [13] é um modelo matemático de processos que torna possível descrever e analisar o comportamento das comunicações efetuadas entre agentes móveis. A representação de mobilidade em π-calculus pode ser expressada através do poder de reconfiguração em tempo real dos agentes, onde a transferência de portas de comunicação entre eles se faz possível. Ele tem sido amplamente utilizado nas especificações de sistemas de agentes móveis [15, 18, 23, 5] e pode ser considerado como base fundamental para as linguagens concorrentes

*Com auxílio à pesquisa do CNPq

com características de mobilidade, da mesma maneira que o λ-calculus é base fundamental para as linguagens funcionais.

Embora existam várias ferramentas de software disponíveis para trabalhar com π-calculus, a grande maioria delas exige do usuário uma infalibilidade em relação à sintaxe: tanto a entrada quanto a saída de dados são difíceis de serem visualizadas, constituindo-se de longas linhas de texto. A conseqüência da dificuldade de especificação de agentes móveis é a desistência do uso de ferramentas principlamente por novos usuários. Além disso, não há um padrão de símbolos adotado por todos os verificadores para π-calculus: cada verificador adota um conjunto próprio de símbolos, o que dificulta ainda mais o uso das várias ferramentas.

Neste trabalho, apresentamos um ambiente integrado para especificação e verificação de agentes em π-calculus constituído pelo editor gráfico PIG (um acrônimo para *Pi-calculus Gráfico*) [6], e pelo verificador VTUBAINA (*A Verification Tool for Up-to Bisimulation and Automata INtegration Automatization*) [1]. Este ambiente tem como objetivo facilitar a especificação de agentes móveis em π-calculus, dado o recurso gráfico de edição, e a verificação de agentes a partir do próprio ambiente gráfico.

2 π-Calculus e a Representação de Agentes Móveis

O π-calculus tem sido amplamente utilizado nas especificações de sistemas de agentes móveis [15, 18, 23, 5]. Ele pode ser considerado como base fundamental para as linguagens concorrentes da mesma maneira que o λ-calculus é base fundamental para as linguagens funcionais.

Suponha os processos S, P e C como um servidor, uma impressora, e um cliente respectivamente [1], onde o servidor S controla o acesso à impressora, a qual o cliente C gostaria de utilizar (conforme mostrado na Figura 1).

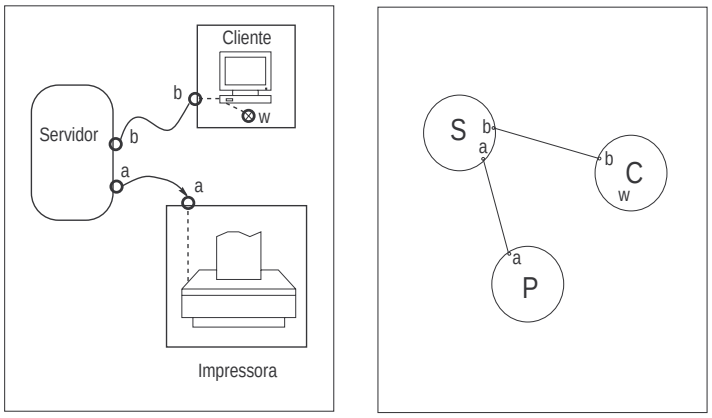


Figura 1: Caracterização dos processos antes da interação. Na direita são mostrados os processos S,P, e C, caracterizando respectivamente o Servidor, Impressora, e Cliente.

Os processos que caracterizam o servidor, cliente, e impressora podem ser especificados em π-calculus, respectivamente, da seguinte forma conforme 2:

$S \stackrel{\text{def}}{=} \overline{b}a.S'$ S tem a capacidade de enviar uma informação (nome a) através da porta b, e passa a se comportar como S'. Neste caso, a informação passada é uma porta de comunicação, isto é, o nome a representa um nome de uma porta de comunicação.

$C \stackrel{\text{def}}{=} b(w).\overline{w}z.C'$ C tem a capacidade de receber uma informação pela porta b, informação referenciada pelo nome w (que neste caso, receberá uma porta de comunicação). Após isto, C tem a capacidade de enviar por w um dado z, passando a se comportar como C'.

$P \stackrel{\text{def}}{=} a(d).P'$ P tem a capacidade de receber uma informação pela porta a que será referenciada por d, e passar a se comportar como P'.

Com o sistema rodando de forma concorrente $(S \mid C \mid P)$ temos:

$$\overbrace{\overline{b}a.S'}^{S} \mid \overbrace{b(w).\overline{w}z.C'}^{C} \mid \overbrace{a(d).P'}^{P}$$

Os processos S e C podem realizar uma comunicação interna entre eles através da porta b, a qual é representada por $\stackrel{\tau}{\longrightarrow}$. Se essa comunicação interna for realizada, então teremos o processo S passando a se

[1] Exemplo retirado de textos introdutórios sobre π-calculus [13, 17]

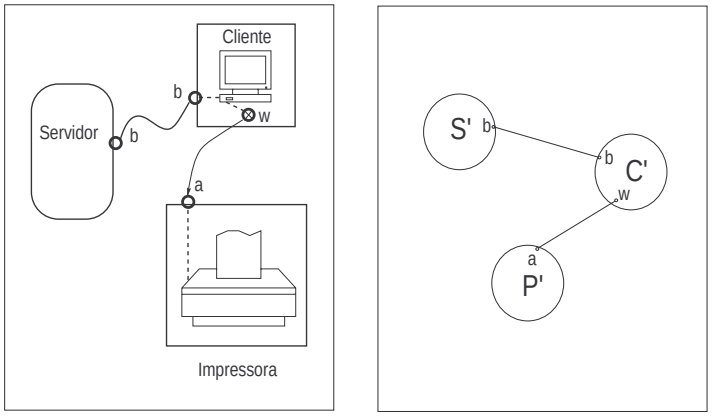


Figura 2: Caracterização dos processos depois da interação.

comportar como S', e em C ocorre a substituição $\{a/w\}$ (substitui-se todos os nomes w por a) [2].

$$(\overline{b}a.S' \mid b(w).\overline{w}z.C' \mid a(d).P') \xrightarrow{\tau} (S' \mid \overline{a}z.C') \mid a(d).P'$$

Após esta comunicação, os processos C e P podem se comunicar através da porta a, ou seja, C pode enviar um dado z para P que irá receber a informação que será referenciada por d em P.

$$(S' \mid \overline{a}z.C' \mid a(d).P') \xrightarrow{\tau} (S' \mid C' \mid P')$$

Em π-calculus, todos os elementos são representados por nomes. Dentro de um conjunto infinito de nomes $\mathcal{N}$, as portas de comunicação, as variáveis e os dados são representados por a,b,c,...,z, os agentes são representados por P,Q,R, e os processos são definidos com a seguinte sintaxe:

$$P ::= \mathbf{O} \mid \alpha.P \mid (\boldsymbol{\nu} x)P \mid (P_1 \mid P_2) \mid (P_1 + P_2) \mid !P$$

Os prefixos $\alpha.P$ podem representar respectivamente uma entrada, uma saída, ou uma ação silenciosa τ: $\alpha ::= a(b) \mid \overline{a}b \mid \tau$

No operador de *Restrição* ($(\boldsymbol{\nu} x)P$) o processo se comporta como P mas o nome x é local, e não pode ser visto pelo ambiente (por outros processos). Como em outras algebras de processos, a restrição $(\boldsymbol{\nu} x)P$ define x sendo um nome local em P (em CCS isto é escrito como $\backslash x$), e desta forma é chamado de *bound name*. Um segundo caso onde um nome pode ser *bound* é quando temos $y(x).P$, onde x é uma variável para qualquer nome que possa ser recebido através do link y. Um nome é chamado de *nome livre* quando ele não é um *nome bound*. Por exemplo, em $\overline{x}z.P$ e em $y(a).Q$, temos que x, z, e y são *livres* pois não dependem de ninguém. O conjunto de nomes livres (*free names*) em α é dado por $\mathbf{fn}(\alpha)$, e $\mathbf{n}(\alpha) = \mathbf{bn}(\alpha) \cup \mathbf{fn}(\alpha)$.

Os operadores $(P_1 \mid P_2)$ e $(P_1 + P_2)$ representam, respectivamente, a composição paralela e a escolha entre os processos. $!P$ representa a replicação do processo P (existem tantas cópias do processo P quanto for necessário - um número ilimitado).

3 Ambiente de Especificação e Verificação

O ambiente de especificação e verificação proposto para π-calculus utiliza, até o momento, um módulo para especifcação gráfica de agentes móveis e um módulo para a verificação de equivalências entre os agentes. Este dois módulos foram inicialmente produzidos como duas ferramentas distintas, PiG e VTubaina, as quais foram posteriormente integradas ao ambiente. Dessa forma, mantivemos os dois módulos com os nomes das ferramentas originais. A idéia é utilizar a PiG para realizar as especificações dos agentes, bem como a visualização gráfica de cada um deles. A verificação é então realizada pela VTubaina, a qual importa o arquivo de especificação gerado a partir de agentes descritos na PiG. A Figura 3 descreve o diagrama de funcionamento do ambiente.

[2] O nome w no processo C se comporta como um nome que servirá como referência ao valor recebido pela porta b. Em CCS [12], somente é permitido que sejam passados valores na comunicação entre os processos, o que não acontece em π-calculus onde é permitia a passagem de portas de comunicação, e processos durante as comunicações.

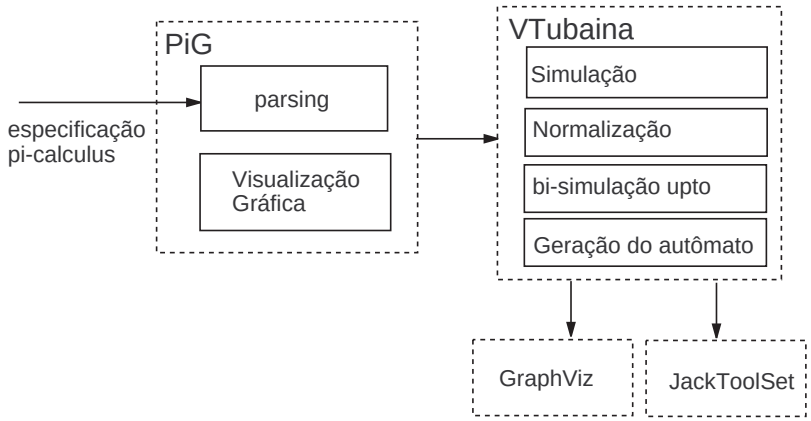


Figura 3: Diagrama do ambiente de especificação e verificação.

3.1 PiG

Apesar de ser um cálculo de referência para agentes móveis e para outros cálculos que tratam mobilidade, os estudos sobre π-calculus se concentram em aspectos teóricos relacionados à expressividade e relações de equivalência computáveis. Desta forma, grande parte das ferramentas desenvolvidas para π-calculus são protótipos que visam a certificação de aspectos teóricos desenvolvidos, sem vislumbrar o usuário final que deseja especificar os seus problemas reais. Por essa razão, alguns problemas são evidentes: a edição texto dos agentes é um processo longo e muitas vezes não intuitivo (principalmente para os novatos); e como a edição é em modo texto, cada ferramenta adota um conjunto de símbolos para representar os operadores π-calculus, e o usuário precisa aprender "uma nova linguagem" para o uso de cada ferramenta.

O módulo PiG - um acrônimo para π-*calculus Gráfico* - tem como objetivo prover um ambiente amigável para especificação de agentes móveis utilizando π-calculus. Neste ambiente, a entrada e a saída de dados podem ser feitas graficamente, agilizando o trabalho do usuário. Como uma árvore sintática é usada internamente, este módulo foi projetado para ser usado juntamente com ferramentas de verificação, funcionando como um *front-end* para as mesmas. Assim, a PiG poderá ser integrado a verificadores existentes para dar suporte à edição gráfica dos agentes em π-calculus. O ambiente apresentado neste artigo é o primeiro passo dessa integração com a VTubaina.

A PiG foi desenvolvida com o objetivo de ser um ambiente altamente configurável e adaptável, podendo lidar ao mesmo tempo com várias representações gráficas do mesmo modelo. Atualmente encontram-se implementadas as Pi-Nets e a linguagem gráfica padrão, que descreveremos e exemplificaremos no decorrer do artigo. Entretanto outras formas de representação gráfica podem ser acopladas à PiG, e a cada modificação da especificação graficamente ou textualmente causa a atualização de todas as outras representações gráficas. Isso permite adotar a representação gráfica que for mais adequada para o tipo de trabalho a ser efetuado.

Existem basicamente duas formas de edição: a introdução direta dos agentes através de uma linguagem textual e a forma gráfica através de uma barra de ferramentas. Essas formas de edição podem ser usadas em conjunto também, sendo que a edição gráfica atualiza a representação textual do processo, e o mesmo ocorre com a edição textual.

Como a PiG é configurável, pode-se criar módulos que se acoplem a ela para permitir ainda novas formas de edição, como por exemplo, editar diretamente a árvore sintática do processo.

A linguagem gráfica utilizada é composta por alguns elementos básicos: ações, processos e operadores. As ações são representadas através de uma caixa e um cone, exceto a ação silenciosa, representada por um pequeno círculo. A representação de um processo é dada pela definição de um processo. O **O** é representado através de um círculo vermelho. Os outros processos através de um quadrado de bordas arredondadas. À exceção da prefixação, todos os outros operadores são representados através de símbolos parecidos com suas representações textuais usuais

O exemplo apresentado, Servidor-Impressora-Cliente, pode ser representado no módulo gráfico como segue:

Como podemos ver na Figura 4, temos a definição de um processo *Press* que é a composição dos processos Servidor, Impressora e Cliente: as duplas barras verticais representam a composição, as ações de entrada e saída levando aos processos *Sl, Cl, Pl*.

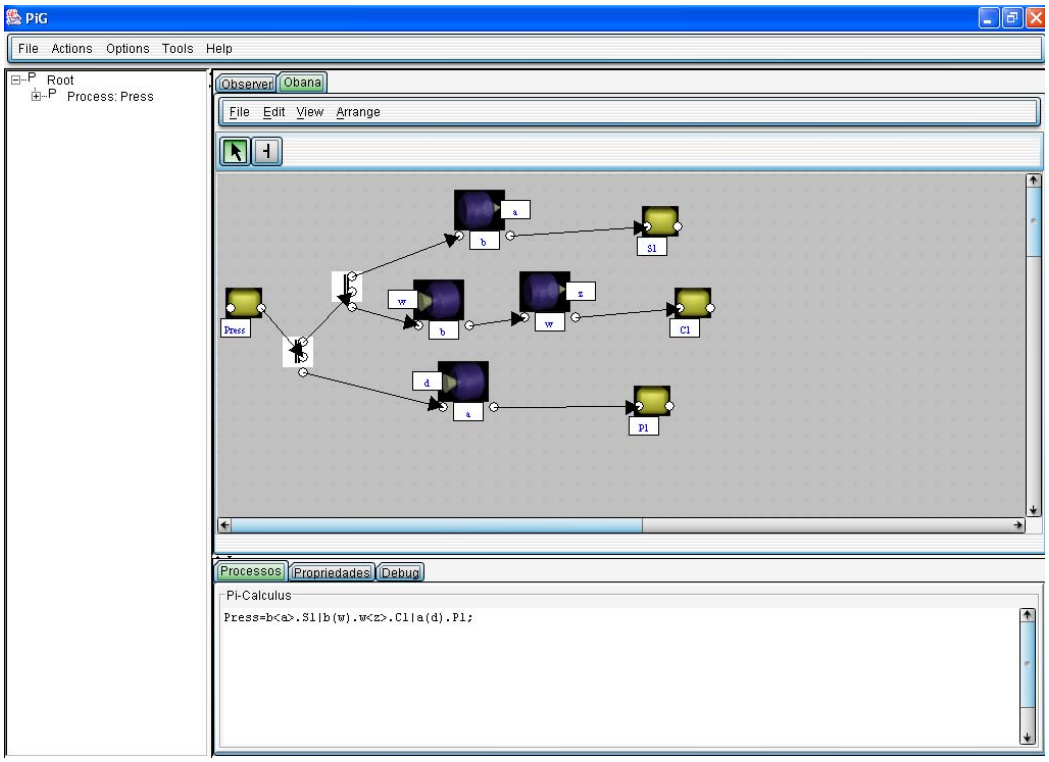


Figura 4: Especificação dos agentes Impressora-Servidor-Cliente

3.1.1 Da PiG para VTubaina

A PiG tem como representação interna uma árvore sintática que agrega a definição de todos os agentes. Através dela, toda a representação gráfica pode ser gerada, bem como toda a representação texto. É a essa árvore que todos os módulos desenvolvidos para a PiG têm acesso e podem modificar.

Para gerar um ambiente capaz de acoplar a PiG à VTubaina, foi necessário apenas fazer com que esse pudesse importar as árvores provenientes da PiG e então gerar os agentes correspondentes descritos na linguagem aceitável pela VTubaina. Assim, em um ambiente gráfico, o usuário pode requisitar a verificação de agentes editados na PiG, e o ambiente gera, de forma transparente ao usuário, os agentes na linguagem da VTubaina e procede a verificação solicitada.

3.2 VTubaina

Para que se possa formalizar como os agentes se comportam, e de alguma forma saber que nível de semelhança existe entre eles, é necessário definir relações de equivalências comportamentais sobre agentes. Na maioria das álgebras de processos [12, 9, 13], a bi-simulação é a idéia matemática mais comum utilizada para estabelecer as equivalências, sob o ponto de vista comportamental, entre processos concorrentes. Em π-calculus, a instanciação de nomes promove um aumento de expressividade no cálculo, mas afeta dramaticamente a teoria, de forma que novas técnicas de verificação precisaram ser desenvolvidas. No presente trabalho, além de implementar algumas técnicas de verificação existentes, implementamos também uma nova técnica de verificação desenvolvida em [1]. VTubaina (*A Verification Tool for Up-to Bisimulation and Automata INtegration Automatization*) é a ferramenta de verificação desenvolvida em [1].

Alguns trabalhos têm sido desenvolvidos para verificação automática em π-calculus [22, 3, 8, 4], e a descoberta de melhorias e novos métodos têm despertado interesse na área de verificação formal em computação móvel. As técnicas atuais para verificação de bi-simulações em π-calculus podem ser subdivididas nas abordagens: semântica e sintática. A abordagem semântica utiliza basicamente os algoritmos de particionamento (*partiton refinement algorithm*) [16, 9], os quais constroem os desdobramentos dos processos (geram todas as possíveis transições), ou os algoritmos baseadas na construção dos sistemas de transição dos agentes à medida que a verificação é realizada (*on-the-fly*) [2]. A abordagem sintática se baseia em métodos de prova *up-to bi-simulação* [19, 20, 8] e é implementada em sistemas de reescritura.

Cada uma das abordagens acima citadas tem suas vantagens: a sintática, na eficiência, e a semântica, na abrangência de agentes com comportamentos semelhantes. Tais diferenças despertou o interesse pelo estudo de uma nova técnica de verificação para π-calculus realizada em [1], a qual utiliza os algoritmos de normal-

ização e de bi-simulação *up-to* (provenientes da abordagem sintática) no auxílio da construção dos autômatos de transição que serão utilizados para verificação (o algoritmo de particionamento - abordagem semântica). Esta nova técnica, bem como as técnicas sintática e semântica, estão implementadas na VTubaina.

3.2.1 Uma nova técnica

Em π-calculus, as duas abordagens de verificação citadas acima têm sido alvo de estudos. A sintática usa a comparação sintática das estruturas dos agentes (expressões), sistemas de normalização e técnicas de prova por bi-simulações *up-to* [19, 20, 8]. Na abordagem semântica, Montanari e Pistore [14] sugerem uma técnica caracterizada pela construção de autômatos de transição dos agentes utilizando algoritmos de refinamento de partições para a verificação de equivalências.

O interesse despertado por essas duas técnicas nos levou a uma proposta de utilização de técnicas de verificação sintática no momento da construção dos desdobramentos dos processos. Devido à ação de entrada poder receber qualquer nome e levar a uma explosão exponencial do número de estados, existe uma preocupação em encontrar processos sintaticamente iguais durante a construção dos autômatos de transição dos agentes. Montanari e Pistore utilizam, para isso, buscas por mapeamento de nomes (*função bijetiva*) e, ao detectar processos semelhantes em estados diferentes, o autômato é reduzido.

Com a utilização apenas de uma busca por processos semelhantes através de um mapeamento de nomes, não conseguimos identificar processos equivalentes e que poderiam estar relacionados em um mesmo estado do autômato. Ao aplicarmos uma verificação sintática baseada em sistemas de normalização e técnicas de prova de bi-simulações *up-to* [19, 20, 8], podemos compactar ainda mais o autômato final. Ao aplicarmos esta técnica durante a verificação entre um processo que está sendo atingido e outros que já foram atingidos no momento da construção do desdobramento das transições do processo, conseguimos detectar um número maior de processos que podem ser agrupados em um mesmo estado do autômato. A diferença de compactação usando a técnica de Montanari e Pistore e a técnica proposta pelo autores [3] será ilustrada no Exemplo 1.

3.2.2 VTubaina: Serviços Fornecidos

A ferramenta VTubaina é o resultado dos estudos sobre técnicas de verificação para π-calculuse, por isso, implementa tanto as técnicas de verificação por refinamento de partições, sistemas de normalização e prova por bi-simulações *up-to*, quanto a nova técnica proposta. A ferramenta efetua as seguintes tarefas:

- Simula as transições de um processo;
- Detecta e visualiza *nomes livres* e *nomes bound* de um processo;
- Aplica apenas o sistema de normalização em um processo, visualizando passo a passo, as regras aplicadas e os resultados gerados na expressão;
- Verifica congruência estrutural entre dois processos;
- Verifica bi-simulações *up-to* entre dois processos;
- Gera o autômato final de transição do processo com desdobramento utilizando apenas os *nomes ativos*;
- Gera o autômato final utilizando a verificação sintática durante os desdobramentos;
- Imprime o autômato final em formato .fc2 (Jack tool Set[7]);
- Imprime o autômato final em formato .dot (dotty - Graph Editor[10, 11]).

A funcionalidade principal deste módulo é de possibilitar a descrição e utilização simultânea das duas abordagens de verificação, tanto semântica (particionamento), quanto sintática (congruência estrutural).

3.2.3 Exemplo de Uso: Geração do Autômato e Verificação

Para ilustrarmos o funcionamento de verificação entre processos descritos em π-calculus no ambiente proposto, especificamos os agentes descritos no Exemplo 1 abaixo no módulo gráfico do ambiente, ilustrado na Figura 5.

Exemplo 1

$$T_1 \stackrel{\text{def}}{=} (\boldsymbol{\nu} a)(a(x).a(x) \mid a(x).\overline{x}b \mid \overline{a}c.a(x) \mid \overline{a}c)$$

$$T_2 \stackrel{\text{def}}{=} (\boldsymbol{\nu} a)(a(x).a(x) \mid a(x).\overline{x}b \mid \overline{a}c.a(x) \mid \overline{a}c.a(x))$$

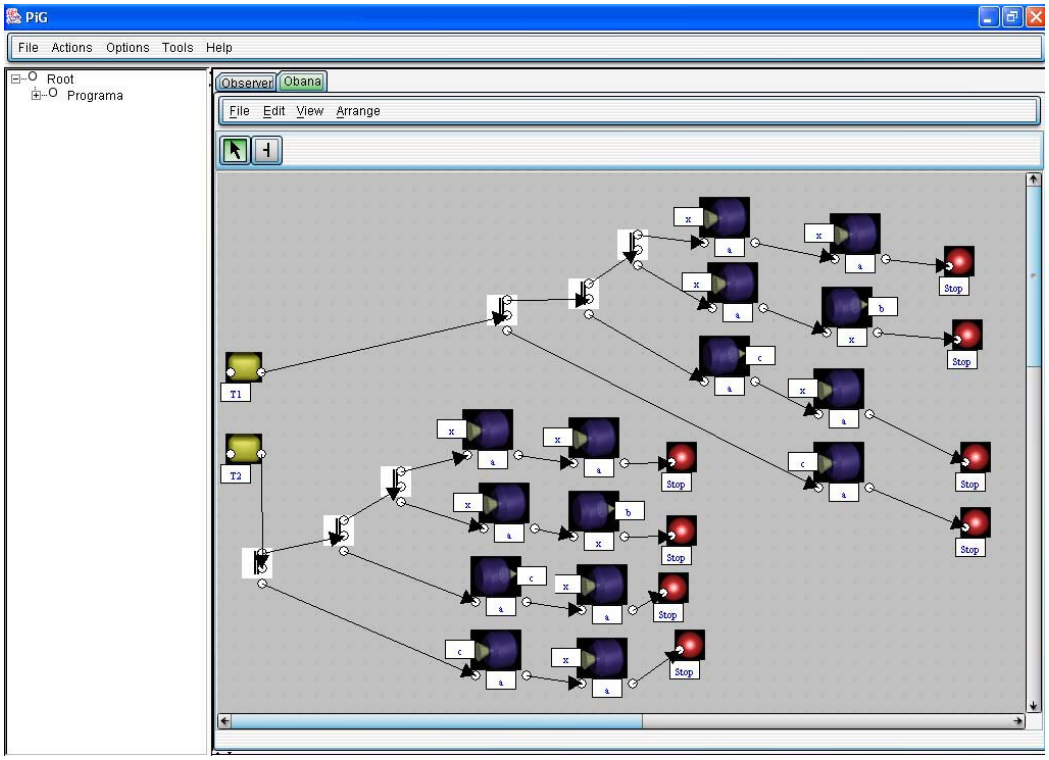


Figura 5: Agentes T_1 e T_2 especificados na PIG

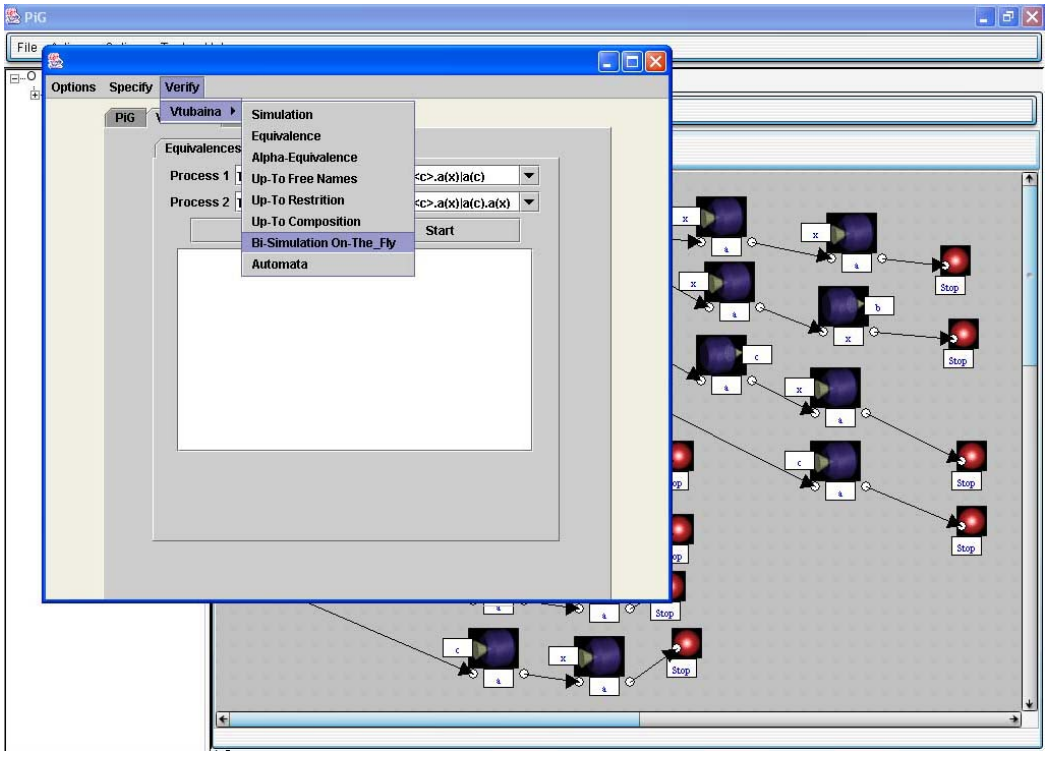


Figura 6: Escolha de Tarefas de Verificação

Após a especificação, exportamos, do módulo PIG, os arquivos `proc_t1.tuba` e `proc_t2.tuba` correspondente à especificação dos processos T_1 e T_2 a serem então carregadas pelo protótipo VTUBAINA. A Figura 6 ilustra os principais serviços fornecidos pelo ambiente para a verificação de agentes móveis.

[3] A nova técnica de verificação está fora do escopo do presente trabalho, detalhes podem ser vistos em [1].

Após especificar os agentes, quando solicitamos a verificação *OnTheFly*, importamos a árvore sintática da PiG, oferecendo os agentes já definidos como opções para a verificação. Após a escolha dos processos, um arquivo no formato aceito pela VTubaina é gerado, contendo os agentes e os comandos solicitados. A VTubaina é então invocada com este arquivo como parâmetro, executando assim a verificação.

No trecho abaixo, temos as linhas de comandos executadas pelo módulo VTubaina, com suas respectivas mensagens.

```
VTUBAINA> loadspec proc_t1.tuba
Agent T1 = (^#0)(#0(#3).#0(#4) | #0(#2).#2<b> | #0<c>.#0(#1) | #0<c>)
Free Names  T1: c,b
Bound Names T1: a,x,x,x,x
Agent T1 was built successfully!

VTUBAINA> loadspec proc_t2.tuba
Agent T2 = (^#0)(#0(#4).#0(#5) | #0(#3).#3<b> | #0<c>.#0(#2) | #0<c>.#0(#1))
Free Names  T2: c,b
Bound Names T2: a,x,x,x,x,x
Agent T2 was built successfully!

VTUBAINA> onthefly T1 T2
Agents <T1> and <T2> are not strong bisimilar!

VTUBAINA> createautomata aut1 T1
Active Names:  c, b
Automata aut1 was built successfully!

VTUBAINA> printautdot aut1 teste1out1
Automata [aut1]: File teste1out1.dot was created successfully!

VTUBAINA> createautomata aut2 T2
Active Names:  c, b
Automata aut2 was built successfully!

VTUBAINA> printautdot aut2 teste1out2
Automata [aut2]: File teste1out2.dot was created successfully!
```

A Figura 7 mostra os autômatos reduzidos gerados pela VTubaina para os agentes T_1 e T_2 do exemplo acima. Para ambos os agentes, o autômato da direita foi gerado com o uso das buscas sintáticas durante os desdobramentos (a nova técnica sugerida), enquanto o da esquerda foi gerado com a aplicação da técnica de particionamento de Montanari e Pistore. Vale salientar que a solicitação da verificação "onthefly" gera apenas os autômatos da direita. Os da esquerda são gerados apenas quando a opção "equivalence" é solicitada (a técnica de particionamento).

Além da verificação de equivalências entre agentes, podemos também gerar a simulação das transições dos processos usando a ferramenta. Neste caso, a simulação é assistida, na qual o usuário designa as informações das ações de entrada dos agentes.

4 Conclusão e Trabalhos Futuros

Neste artigo, apresentamos um ambiente para especificação e verificação que pode ser utilizado como ferramenta para auxiliar a manipulação de agentes móveis modelados utilizando o cálculo de processos π-calculus. Este cálculo foi primordialmente proposto como teoria base para agentes móveis, e não como linguagem de especificação. Apesar de ser um abuso de notação, existe um grande uso da teoria na especificação como forma de ilustrar a verificação de agentes. Desta forma, o ambiente proposto se presta mais para fins didáticos, com o objetivo de ilustrar a especificação e verificação de agentes móveis, do que para fins industriais. Para tais objetivos, linguagens de especificação baseadas tanto em π-calculus quanto em outras teorias têm sido propostas.

A construção do protótipos nos trabalhos de pesquisa científica constituí uma parte fundamental durante um estudo. Geralmente, protótipos são criados para obtenção de novos resultados e melhor visualização da automação do processo de verificação. Isso tem ocasionado na não reutilização de grande parte dos códigos encontrados nos vários trabalhos de pesquisa realizados em π-calculus por diferentes cientistas. Neste caso, tornaria-se interessante uma biblioteca contendo um ambiente de manipulação sintática e de simulação de processos descritos em π-calculus no intuito de ajudar a confecção de novos protótipos para estudos nessa área.

Inicialmente, o ambiente aqui proposto utiliza duas ferramentas, entretanto a junção de novos verificadores na construção de um único ambiente é de vital importância. Temos como objetivo acoplar novos verificadores a este ambiente gráfico de forma que o usuário possa editar seus agentes em um único ambiente, e ao mesmo tempo, utilizar as facilidades dos vários verificadores.

Como trabalho futuro também temos o desenvolvimento de novos componentes para a PiG, capazes de exibir e editar processos através de outros tipos de representações gráficas, ampliando a comunidade de usuários e a versatilidade da ferramenta.

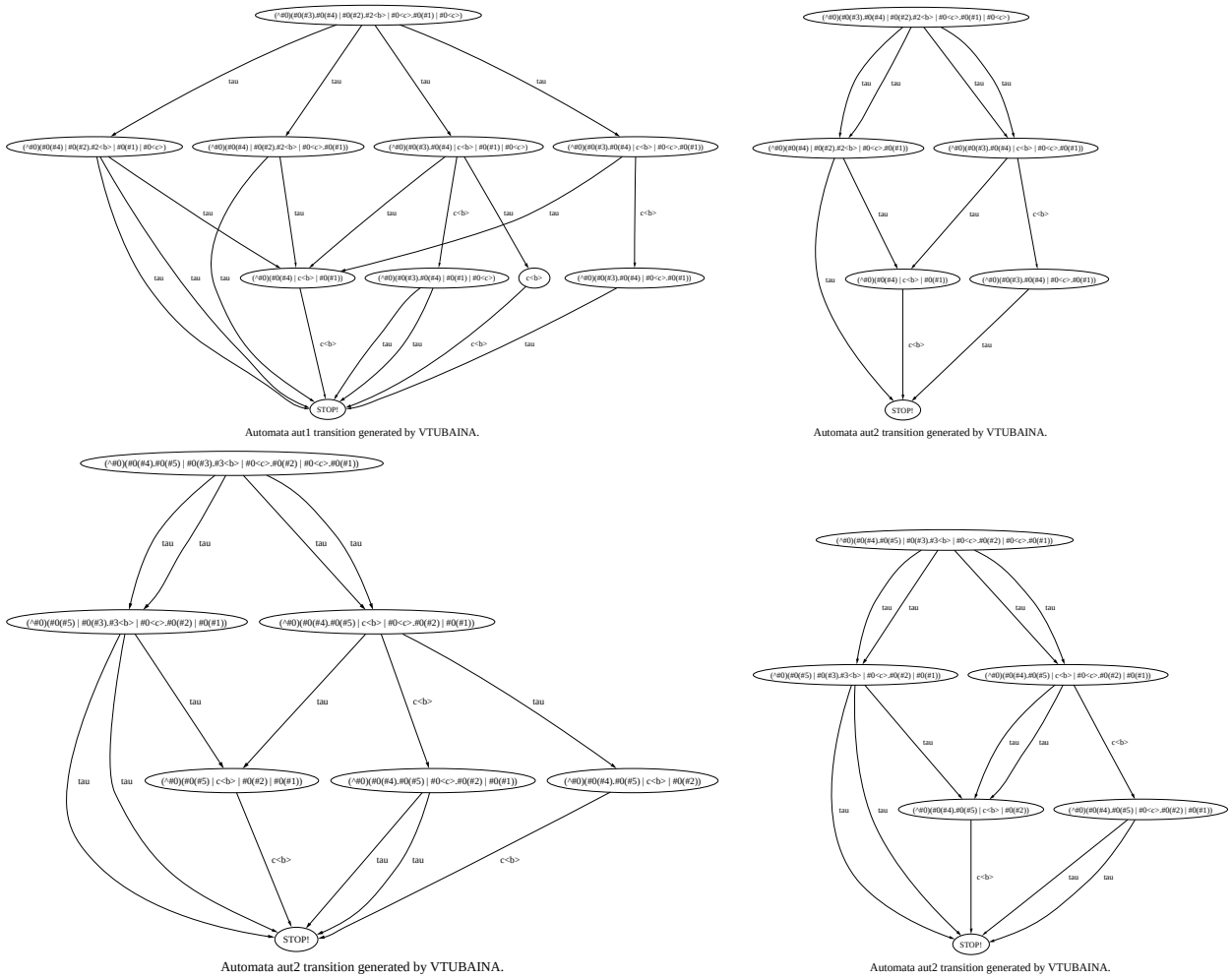


Figura 7: Autômatos para T_1 e T_2 construídos pela VTubaina

Outra possível aplicação seria o desenvolvimento de componentes capazes de integrar a PiG e o próprio Pi-Calculus com o processo de verificação de software adotado nas empresas que fazem uso de verificação formal. Este tipo de medida permitiria que o ambiente deixasse de ter apenas fins didáticos para ter uma real aplicação industrial.

Referências

[1] M. M. Amorim. Uma técnica de verificação para π-calculus baseada em bi-simulação up-to e algoritmos de particonamento. Master's thesis, IME/USP - Instituto de Matemática e Estatística da Universidade de São Paulo, 2003.

[2] J-C. Fernandez and L. Mounier. "on the fly" verification of behavioural equivalence and preorders. In *Proc. 3rd International Computer Aided Verification Conference*, pages 181–191, 1991.

[3] G. Ferrari, S. Gnesi, U. Montanari, M. Pistore, and G. Ristori. Verifying mobile processes in the HAL environment. In Alan J. Hu and Moshe Y. Vardi, editors, *Proceedings of CAV '98*, volume 1427 of *LNCS*. Springer, 1998. Tool Poster.

[4] G. Ferrari, G. Modoni, and P. Quaglia. Towards a Semantics-Based Environment for the π-Calculus. In *Proceedings of 5th Italian Conference on Theoretical Computer Science (ICTCS-95)*. World Scientific, 1995.

[5] C. Fournet and G. Gonthier. The reflexive chemical abstract machine and the join-calculus. In *Proceedings of POPL '96*, pages 372–385. ACM, January 1996.

[6] Andrade. Andre G. Pig - pi calculus grafico. *Atas do XVII Colóqui de Iniciação Científica*, pages 1–8, 2003.

[7] S. Gnesi. A formal verification environment for concurrent systems design. In *Proceedings Workshop on Automated Formal Methods*, volume 5 of *ENTCS*. University of Oxford, 1996.

[8] D. Hirschkoff. Automatically proving up to bisimulation. In Petr Jancar and Mojmir Kretinsky, editors, *Proceedings of MFCS '98 Workshop on Concurrency*, volume 18 of *ENTCS*. Elsevier Science Publishers, 1998.

[9] P. C. Kanellakis and S. A. Smolka. CCS expressions, finite state processes, and three problems of equivalence. *Information and Computation*, 86(1):43–68, May 1990.

[10] E. Koutsofios. Editing graphs with *dotty*. Technical report, AT&T Bell Laboratories, Murray Hill, NJ, USA, July 1994. This report, and the program, is included in the `graphviz` package, available for non-commercial use at `http://www.research.att.com/sw/tools/graphviz/.`

[11] E. Koutsofios and S. North. Drawing graphs with *dot*. Technical Report 910904-59113-08TM, AT&T Bell Laboratories, Murray Hill, NJ, USA, September 1991.

[12] R. Milner. *Communication and Concurrency*. Prentice-Hall, Englewood Cliffs, New Jersey, 1989.

[13] R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, part I/II. *Journal of Information and Computation*, 100:1–77, September 1992.

[14] U. Montanari and M. Pistore. Checking bisimilarity for finitary π-calculus. In Insup Lee and Scott A. Smolka, editors, *Proceedings of CONCUR '95*, volume 962 of *LNCS*, pages 42–56. Springer, 1995.

[15] F. Orava and J. Parrow. An algebraic verification of a mobile network. *Journal of Formal Aspects of Computing*, 4:497–543, 1992.

[16] R. Paige and R. E. Tarjan. Three partition refinement algorithms. *SIAM Journal on Computing*, 16(6):973–989, December 1987.

[17] J. Parrow. An introduction to the π-calculus. In Bergstra, Ponse, and Smolka, editors, *Handbook of Process Algebra*, pages 479–543. Elsevier, 2001.

[18] D. Sangiorgi. *Expressing Mobility in Process Algebras: First-Order and Higher-Order Paradigms*. PhD thesis, LFCS, University of Edinburgh, 1993. CST-99-93 (also published as ECS-LFCS-93-266).

[19] D. Sangiorgi. On the bisimulation proof method. *Mathematical Structures in Computer Science*, 8(5):447–479, 1998. An extended abstract appeared in the *Proceedings of MFCS '95*, LNCS 969: 479–488.

[20] D. Sangiorgi and R. Milner. The problem of "weak bisimulation up to". In W. R. Cleaveland, editor, *CONCUR '92: Third International Conference on Concurrency Theory*, volume 630 of *Lecture Notes in Computer Science*, pages 32–46, Stony Brook, New York, 24–27August 1992. Springer-Verlag.

[21] B. Thomsen. A theory of higher order communicating systems. *Journal of Information and Computation*, 116(1):38–57, 1995.

[22] B. Victor. *A Verification Tool for the Polyadic π-Calculus*. Licentiate thesis, Department of Computer Systems, Uppsala University, Sweden, May 1994. Available as report DoCS 94/50.

[23] D. Walker. Objects in the π-calculus. *Journal of Information and Computation*, 116(2):253–271, 1995.

Estimativas por Tipo de Produto de Trabalho: uma Extensão da técnica PCU para o CMMI-SW Nível 2

Tatiana Cavalcanti Monteiro
Universidade de Fortaleza – UNIFOR, MIA
Fortaleza, Ceará, Brasil
tatiana.m@uol.com.br

e

Carlo Giovano S. Pires
Instituto Atlântico, Dept. SPD
Fortaleza, Ceará, Brasil
cgiovano@atlantico.com.br

e

Arnaldo Dias Belchior
Universidade de Fortaleza – UNIFOR, MIA
Fortaleza, Ceará, Brasil
belchior@unifor.br

Resumo

O CMMI-SW nível 2 recomenda a implantação de atividades de estimativas de tamanho, esforço, prazo e custo, como forma de melhorar o planejamento e o acompanhamento de projetos de software. Apesar de haver várias técnicas de estimativas, a utilização das mesmas em empresas de software ainda não é uma prática tão comum. A técnica PCU (Pontos por Caso de Uso), por exemplo, tem-se mostrado adequada para produtos de software orientados a objetos e baseados em casos de uso. No entanto, a granularidade dessa técnica para produtos de trabalho gerados nas atividades de planejamento e acompanhamento não se tem mostrado muito adequada. Este trabalho apresenta uma extensão da técnica PCU, para atender recomendações do CMMI-SW nível 2, permitindo uma visão mais detalhada das estimativas por tipo de produto de trabalho, possibilitando realizar refinamentos dessas estimativas ao longo do processo de desenvolvimento.
Palavras-chave: Estimativas, PCU, Casos de Uso, CMMI-SW.

Abstract

The CMMI-SW level 2 requires estimation's use, including size, effort, schedule and cost, like the best way to improve software projects planning and tracking. Therefore there are many techniques for software estimations; this use isn't a common practice in software organizations. The UCP (Use Case Points), for example, is greatly suitable to object-oriented software and based in use case model. Nevertheless, the granularity offered by this method to the work's products types for software projects planning and tracking hadn't been adequate. This work presents an extension of UCP technique to attend CMMI-SW level 2, allowing detail view about the estimation by work's products types, offering to refine these estimations across the software development process.
Keywords: Estimation, UCP, Use Cases, CMMI-SW.

1. INTRODUÇÃO

Uma das práticas exigidas pelo CMMI-SW (*Capability Maturity Model Integration for Software*) nível 2 [2] é a realização de estimativas para tamanho, esforço, prazo e custo. Uma das grandes dificuldades na implantação das práticas de estimativa é a utilização de técnicas adequadas às exigências do modelo.

Algumas organizações têm adaptado técnicas de estimativas às suas necessidades, gerando técnicas proprietárias para atender a essas práticas, dificultando a troca de experiências, o uso de bases de conhecimentos de outras organizações, além do esforço despendido e do custo de definição dessas técnicas proprietárias.

As estimativas apóiam essencialmente as atividades de planejamento e acompanhamento de projetos de software. Estimativas eficientes permitem a verificação da viabilidade do projeto, a elaboração de propostas técnicas e comerciais, a confecção de planos e cronogramas detalhados, e o acompanhamento efetivo de projetos.

As estimativas podem ser enquadradas em dois grupos: *bottom-up* e *top-down*. As estimativas *bottom-up* são utilizadas para calcular cada item de trabalho, depois sumarizá-los ou agregá-los para se obter a estimativa total do projeto. Podem ser realizadas através da opinião de algum especialista ou através de uma base de dados, para determinar a complexidade e o esforço associados a uma determinada tarefa. As estimativas *top-down* avaliam o projeto como um todo, utilizando técnicas especializadas. A granularidade das estimativas dos produtos de trabalho é determinada pela estimativa de todo o projeto. Informações de projetos análogos são em geral utilizadas como base para a estimativa *top-down*.

Dentre as diversas técnicas para se estimar projetos de software, algumas delas tem-se consolidado como: Análise por Ponto de Função (*Functional Point Analysis* – FPA) [1], MKII *Function Points* [1] [9] e Pontos por Casos de Uso [11]. Esta última baseou-se nas duas primeiras técnicas e, apesar de ser uma técnica ainda recente, tem sido aprimorada e tem apresentado crescente utilização na indústria. A PCU é do tipo top-down e adequada a projeto que descrevem seus requisitos de software através de casos de uso.

Este trabalho apresenta uma extensão da técnica PCU, para atender recomendações do CMMI-SW nível 2 [2] – a TUPC (*Technical Use Case Point*) – permitindo uma visão mais detalhada das estimativas por tipo de produto de trabalho, possibilitando realizar refinamentos dessas estimativas ao longo do processo de desenvolvimento. Além disso, a proposta propõe calibrações nos fatores de produtividade por tipo de produto de trabalho, permitindo melhorar essas estimativas.

Este trabalho está organizado em cinco seções. Na seção 2, descreve-se como as estimativas são apresentadas no CMMI-SW nível 2. Na seção 3, delineia-se a técnica PCU. Na seção 4, discorre-se sobre a extensão da PCU proposta, a TUCP. Na seção 5, apresenta-se um estudo de caso, utilizando-se a TUCP. Finalmente, na seção 6, são delineadas as conclusões do trabalho.

2. ESTIMATIVAS NO CMMI-SW NÍVEL 2

O CMMI-SE é um modelo para avaliação e melhoria da maturidade dos processos de uma organização, como uma integração e evolução dos modelos: SW-CMM (*Capability Maturity Model for Software*), SECM (*System Engineering Capability Model*), e IPD-CMM (*Integrated Product Development CMM*) [2].

O CMMI-SW fornece às organizações de software um guia de como obter controle em seus processos para desenvolver e manter software, e como evoluir em direção a uma cultura de engenharia de software e excelência de gestão. O CMMI foi projetado para guiar as organizações de software no processo de seleção das estratégias de melhoria, determinando a maturidade atual do processo e identificando as questões mais críticas para a qualidade e melhoria do processo de software [2].

O CMMI tendo em vista a realidade e as necessidades das organizações, e visando fornecer uma maior flexibilidade, admite duas abordagens: uma por estágio, como o CMM tradicional, e outra contínua, semelhante a ISO/IEC 15504 [3]. A representação por estágios contém cinco níveis de maturidade: 1-inicial, 2-gerenciado, 3-definido, 4-gerenciado quantitativamente, e 5-otimizado.

O CMMI é organizado em cinco níveis de maturidade que definem uma escala ordinal para medir a maturidade de um processo de software da organização e para avaliar a sua capacidade. Os níveis também ajudam uma organização a priorizar seus esforços de melhorias. Cada nível compreende um conjunto de objetivos de processos que, quando satisfeitos, estabilizam um componente importante do processo de software. Alcançando cada nível da estrutura de maturidade, estabelecem-se diferentes componentes no processo de software, resultando em um crescimento na capacidade de processo da organização.

A representação por estágio do CMMI-SW possui as seguintes áreas de processo para o nível 2 (gerenciado): gerência de requisitos, planejamento de projeto, monitoração e controle de projeto, gerencia de acordo com fornecedores, medição e análise, garantia da qualidade do processo e do produto, e gerência de configuração. A representação contínua do CMMI-SW possui seis níveis de maturidade: incompleto, executado, gerenciado, definido, gerenciado quantitativamente, e otimizado [2].

As áreas de processo do CMMI-SW podem ser agrupadas em quatro categorias: gerenciamento de processo, gerenciamento de projeto, engenharia, e suporte. As áreas de processos relacionadas à gerência de projetos contêm as atividades de projetos referentes ao planejamento, acompanhamento e controle do projeto. São elas: planejamento de projeto, acompanhamento e controle do projeto, gestão de acordos com fornecedores, gestão integrada de projeto, gestão de riscos, integração da equipe, gestão quantitativa de projeto, e gestão integrada de fornecedores [2].

No CMMI-SW, como também no SW-CMM [2], o processo de estimativas estabelece uma base formal para o planejamento e o acompanhamento de projetos de software, a partir de quatro atividades: (i) estimar o tamanho do produto a ser gerado; (ii) estimar o esforço empregado na execução do projeto; (iii) estimar a duração do projeto; e (iv) estimar o custo do projeto. Essas atividades são referenciadas no CMMI-SW nível 2 [2], nas áreas de processo de planejamento de projeto (*Project Planning* - PP) e monitoração e controle de projeto (*Project Monitoring and Control* - PMC) .

No CMMI-SW nível 2, as estimativas de tamanho devem ser feitas para todas as principais atividades e produtos de software, e realizadas sobre produtos com granularidade fina e adequada para um acompanhamento contínuo. No entanto, este modelo não especifica o tipo de medição que deve ser aplicada, nem a granularidade em que os produtos de trabalho devem ser decompostos. Para que um projeto seja melhor estimado, seus produtos de trabalho devem ser decompostos até uma granularidade necessária para se alcançar os objetivos da estimativa [2].

O CMMI-SW exige que se planeje e acompanhe as atividades do processo de desenvolvimento de software, através da adoção de métodos e modelos de ciclo de vida que guiem as atividades de desenvolvimento, a documentação dos produtos de trabalho e o planejamento [2].

O planejamento deve englobar todas as atividades do desenvolvimento, além de estabelecer indicadores e métricas de qualidade. Esses indicadores permitem acompanhar o desempenho real dos projetos, comparando-os aos seus valores estimados e planejados. Nos casos em que os resultados reais divergem dos planejados, ações corretivas devem ser tomadas e gerenciadas até a sua efetiva conclusão.

A seguir, será apresentada a técnica de pontos por casos de uso (PCU), que será utilizada como base para o planejamento e acompanhamento de projetos em organizações focadas no CMMI-SW.

3. PONTOS POR CASOS DE USO (PCU)

Os projetos de software orientado a objetos já utilizam, comumente, os diagramas de casos de uso (use cases) para descrever as funcionalidades do sistema de acordo com a forma de utilização por parte dos usuários. Tomando como base os casos de uso, uma técnica para estimativa de projeto foi proposta em 1993 por Gustav Karner [4]. Essa técnica permite estimar o tamanho do sistema ainda na fase de levantamento de casos de uso, utilizando-se dos próprios documentos gerados como subsídio para o cálculo dimensional.

Apesar de ser uma técnica recente, o PCU tem sido estudado por vários pesquisadores no meio acadêmico e na indústria. Em [11], os resultados da aplicação de PCU na estimativa de alguns projetos comerciais são relatados.

Para essa técnica ser eficaz, os casos de uso devem ser descritos em um nível de detalhamento apropriado, pois isto influência diretamente na qualidade final da estimativa. Portanto, deve haver a preocupação em medir casos de uso em nível de sistema – onde seja possível diferenciar transações e interações com o usuário. Desta forma, casos de uso de nível muito alto (modelagem de negócio do sistema) ou muito baixo (decomposição funcional dos casos de uso) não se demonstram adequados para a medição.

Várias pesquisas têm sido realizadas nos últimos anos, discutindo o nível de decomposição ideal dos casos de uso para a aplicação da técnica. Este é o principal problema da técnica PCU: na maioria dos casos, caberá aos analistas decidir a granularidade ideal dos casos de uso utilizados para a medição. Uma técnica de descrição dos casos de uso muito utilizada em [8] apresenta como deve ser representado um requisitos através de um caso de uso.

Uma vez que os casos de uso principais do sistema sejam levantados e descritos, é possível estimar o tamanho total do software baseando-se em um conjunto simples de métricas. As atividades necessárias para a geração da estimativa baseada na técnica PCU são decompostas nas seguintes atividades:

- Classificar os atores;
- Classificar os casos de uso;
- Definir os fatores técnicos e de ambiente.

A primeira atividade é classificar os atores envolvidos em cada caso de uso, obtendo-se um somatório dos pontos não-ajustado. A classificação desses atores está baseada na Tabela 1. O peso total dos atores do sistema (*Unadjusted Actor Weight* - UAW) é calculado pela soma dos produtos do número de atores de cada tipo pelo respectivo peso.

Tabela 1 - Classificação do Atores

Tipo de Ator	Descrição	Peso
Simples	Aplicação com API definida	1
Intermediário	Outro sistema interagindo através de um protocolo de comunicação, como TCP/IP ou FTP	2
Complexo	Um usuário interagindo através de uma interface gráfica (*stand-alone* ou Web)	3

Uma vez calculado o peso dos atores do sistema, a atividade seguinte é classificar os casos de uso e calcular o peso bruto dos mesmos (*Unadjusted Use Case Weight* - UUCW). Para fins de cálculo, dividem-se os casos de uso em três níveis de complexidade, de acordo com o número de transações envolvidas em seu processamento.

Por transação, entende-se como uma série de processos que devem ser realizados em conjunto ou cancelados em sua totalidade, caso não seja possível concluir o processamento. A Tabela 2 apresenta o peso para cada um dos tipos de caso de uso classificados.

Tabela 2 - Classificação dos Casos de Uso

Tipo de Caso de Uso	Número de Transações	Peso
Simples	Até 3 transações	1
Intermediário	De 4 a 7 transações	2
Complexo	Acima de 7 transações	3

O cálculo do UUCW é realizado de forma similar ao cálculo do peso dos atores, somando os produtos da quantidade de casos de uso classificados em cada tipo pelo peso nominal do tipo em questão.

Para calcular os pontos de casos de uso não ajustados (*Unadjusted Use Case Points* - UUCP) apresentados na Eq. (1), deve-se somar o valor obtido na medição de atores com o valor obtido na medição de casos de uso.

$$UUCP = UAW + UUCW \qquad \textbf{Eq. (1)}$$

A atividade seguinte trata de definir os fatores técnicos (*Technical Complexity Factor* – TCF) e o de ambiente (Environmental Factors - EF). Os fatores técnicos (TCF) medem a complexidade do projeto em relação aos requisitos não-funcionais. Esses fatores técnicos influenciam no resultado do PCU, apresentados em Eq.(2). Segundo Karner [4], os fatores de complexidade do projeto são as características relacionadas à performance, portabilidade, segurança, reusabilidade de código, entre outras.

Já os fatores de ambiente (EF) estão relacionados com a familiaridade com o processo de desenvolvimento a ser utilizado no projeto, a experiência na aplicação, motivação, requisitos estáveis, entre outros.

Para calcular o fator de complexidade técnica do sistema (TCF), deve-se primeiro calcular o TFactor. O TFactor é o somatório do produto entre o valor atribuído no projeto para cada item e seu peso. O TCF é calculado por:

$$TCF = 0{,}6 + (0{,}01 * TFactor) \qquad \textbf{Eq. (2)}$$

Para calcular os fatores de ambiente - EF apresentado em Eq.(3) deve-se primeiro calcular o EFactor. O EFactor é o somatório do produto entre o valor atribuído no projeto para cada item e seu peso [11]. O EF é calculado por:

$$EF = 1{,}4 + (-0{,}03 * EFactor) \qquad \textbf{Eq. (3)}$$

Os pontos de caso de uso (PCU) são calculados por:

$$PCU = UUCP * TCF * EF \qquad \textbf{Eq. (4)}$$

O Esforço apresentado em Eq.(5) é obtido pelo produto do tamanho em PCU pela produtividade, que é calculado por:

$$\text{Esforço} = \text{PCU} * \text{Produtividade} \qquad \textbf{Eq. (5)}$$

As contagens de PCU em Eq.(4) podem variar entre organizações e indivíduos, devido à mencionada variação nos estilos de casos de uso. É então razoável supor que a produtividade associada ao desenvolvimento de 1 PCU (20 homens-horas, no trabalho original de Karner [4]) também varie bastante. Desta forma, a obtenção de estimativas confiáveis de esforço exigiria a padronização dos estilos de casos de uso em um extenso trabalho de calibração do modelo de estimativas baseado em PCU.

A seguir, será desenvolvida a abordagem para estimativas de produtos de trabalho baseado em PCU, a TUCP no contexto do CMMI-SW.

4. UMA ABORDAGEM PARA ESTIMATIVAS DE PRODUTOS DE TRABALHO BASEADA EM PCU

As estimativas podem ser realizadas em vários pontos do ciclo de vida, e em um subconjunto dos principais produtos de trabalho, dependendo de características e estratégias do projeto. Quanto aos pontos do ciclo de vida, é possível realizar as estimativas na fase de proposta com requisitos de alto nível e heurísticas [11], depois refinar após detalhamento de requisitos, ou refinar após a especificação de casos de uso, ou até mesmo após análise e projeto.

Conforme descrito na seção 2, o CMMI-SW nível 2 não define quais são os produtos de trabalho e nem o nível de granularidade necessário para se alcançar os objetivos das estimativas. As empresas são responsáveis por determinar esse nível de granularidade em seus produtos de trabalho de acordo com os objetivos e as estratégias necessárias de cada organização.

A técnica original PCU apresenta a estimativa de tamanho para todo o projeto em PCU e a estimativa de esforço pode ser obtida através de fatores de produtividade. No entanto, esta granularidade não permite um planejamento detalhado e nem um acompanhamento efetivo por produtos de trabalho.

Neste contexto, este trabalho propõe uma extensão da técnica PCU, para atender recomendações do CMMI-SW nível 2, permitindo uma visão mais detalhada das estimativas por tipo de produto de trabalho, possibilitando realizar refinamentos dessas estimativas ao longo do processo de desenvolvimento. Essa extensão, a TUCP (Technical Use Case Point) compreende os quatro pontos a seguir:

- Estimativa de tamanho do projeto;
- Estimativas de tamanho por produtos de trabalho;
- Estimativa de esforço com fator de produtividade diferenciado;
- Estimativas de prazo e custo.

Os quatro pontos propostos pela técnica TUCP serão apresentados abaixo, juntamente com um exemplo de utilização da mesma.

4.1 Estimativa de Tamanho do Projeto

Pontos de casos de uso ajustados (PCU) apresentados em Eq.(4) consideram o peso dos diferentes tipos de interface através dos atores, o peso dos requisitos funcionais através dos casos de uso, o peso dos requisitos não-funcionais através dos fatores técnicos, e o peso da equipe e ambiente de desenvolvimento através dos fatores de ambiente. Um problema identificado com esta abordagem é que o uso dos fatores de ambiente - EF pode levar a diferentes tamanhos, dependendo da equipe ou empresa que desenvolver o projeto. Apenas os fatores técnicos e não-técnicos deveriam ser levados em consideração para a estimativa de tamanho, evitando gerar dependência do tamanho do software com características de empresa e equipe de desenvolvimento. De acordo com a proposta desse trabalho, os fatores de ambientes devem ser considerados apenas na estimativa de esforço.

A técnica proposta define a medida TUCP (*Technical Use Case Point*) para obtenção de um valor de tamanho de projeto baseado apenas nos requisitos funcionais e nos requisitos não-funcionais do sistema. Essa medida é obtida pelo ajuste do UUCP apenas pelos fatores técnicos - TCF.

$$\text{TUCP} = \text{UUCP} * \text{TCF} \qquad \textbf{Eq. (6)}$$

4.2 Estimativas de Tamanho por Produtos de Trabalho

Para a realização de estimativas de produtos de trabalho, deve-se inicialmente definir os tipos de produto. De acordo com os modelos SW-CMM [6][7]e CMMI-SW [2], um projeto de desenvolvimento de software trabalha com quatro tipos principais de produtos de engenharia: Requisitos, Análise e Projeto, Codificação, e Teste. Um tipo de produto de trabalho ainda pode ser categorizado em subtipos.

Por exemplo, para o tipo de produto de trabalho do tipo Análise e Projeto, podemos ter os subtipos: projeto gráfico, especificação de arquitetura, projeto detalhado, especificação de banco de dados, especificação de IHC (Interface Humano-Computador) para cada caso de uso. Desta forma, a granularidade torna-se mais fina para estimar os casos de uso, sendo mais fácil o acompanhamento do projeto de software e permitindo ações corretivas antes do final de toda a análise e projeto.

Com base nos tipos de produto, as estimativas de tamanho podem ser realizadas para cada produto de caso de uso. Por exemplo, um projeto com o caso de uso Autenticar Usuário e os tipos de produtos Requisitos, Análise e Projeto, Codificação e Teste poderia ter os seguintes produtos de trabalho:

- Especificação do Caso de Uso Autenticar Usuário: Descrição, Regras de negócio, fluxos, pré-condições, pós-condições, entre outras seções.
- Projeto do Caso de Uso Autenticar Usuário: Realização do caso de uso com diagramas de classe e seqüência para implementação do caso de uso.
- Código do Caso de Uso Autenticar Usuário: Classes definidas no projeto do caso de uso e implementadas na linguagem definida.
- Teste do Caso de Uso Autenticar Usuário: Casos de teste para os cenários de execução.

De acordo com a técnica PCU, o esforço do projeto é diretamente proporcional ao seu tamanho em pontos de caso de uso. Assim sendo, o tamanho de um produto de caso de uso pode ser baseado no percentual de esforço empregado para seu desenvolvimento (ver exemplo de medições de percentual de esforço por tipo de produto na Tabela 3).

Medições realizadas para o projeto, ou por tipo de projeto podem ser utilizadas para determinar um percentual de distribuição adequado para a instituição. O tamanho de cada produto de caso de uso é então definido de forma proporcional a seu fator de complexidade e o seu percentual de distribuição de esforço por tipo de produto.

$$\text{TUCP(produto de caso de uso)} = \text{Peso do Caso de Uso} / (\text{Soma de Todos os Pesos de Casos de Uso})) * \text{TUCP} * \text{PEP} \qquad \textbf{Eq. (7)}$$

O tamanho do produto de trabalho por caso de uso definido em Eq.(7) é dado pelo produto do resultado da divisão entre o peso do caso de uso sobre a soma do peso de todos os casos de uso do sistema, com o tamanho de todo o sistema - TUCP na Eq.(6) e com o percentual de esforço por produto - PEP (Veja Tabela 3).

4.3 Estimativa de Esforço com Fator de Produtividade Diferenciado

Para obter a estimativa de esforço, o método PCU utiliza um fator de multiplicação (ou produtividade) comum para todos os tipos de atividade de um projeto. No entanto, a produtividade pode ser variável, dependendo do tipo de projeto e da equipe que realiza cada tipo de atividade. Por exemplo, em uma organização que possui *frameworks* ou componentes reutilizáveis, a produtividade de atividades de codificação costuma ser alta.

No entanto, as atividades de "especificação de requisitos" e "análise e projeto" não serão agilizadas devido ao reuso de código. Portanto, a organização deverá, a partir das medições de esforço realizado para cada tipo de atividade e os TUCP´s calculados para os projeto, manter uma base de dados com fator de produtividade para cada tipo de atividade e característica de projeto.

O Esforço para gerar um produto de caso de uso é calculado por:

$$\text{Esforço} = \text{TUCP (produto de caso de uso)} * \text{EF} * \text{FPP} \qquad \textbf{Eq. (8)}$$

O Esforço definido em Eq.(8) gasto em um produto de trabalho por caso de uso é dado pelo produto do tamanho do caso de uso - TUCP apresentado em Eq.(7) com o fator de ambiente do sistema – EF apresentado em Eq.(3) e com o fator de produtividade do tipo de produto de trabalho - FPP(Veja Tabela 4).

4.4 Estimativas de Prazo e Custo

As estimativas de prazo e custos devem ser realizadas a partir do esforço estimado com a técnica proposta, disponibilidade de recursos, restrições do projeto e custos homens-horas. Note que essa estimativa pode ser feita para o projeto como um todo ou para cada produto de trabalho através do esforço estimado para o produto.

4.5 Exemplo de TUCP

Para exemplificar a técnica, vamos considerar o caso de uso Autenticar Usuário como intermediário, com uma interface gráfica (ator Complexo) em uma empresa que possui medições e características de projeto de acordo com a Tabela 3 e Tabela 4, uma equipe produtiva e um fator técnico de complexidade mediano. Vamos considerar também que existe o caso de uso Cadastrar Usuário de característica Simples.

Tabela 3 - Percentual de esforço por produto - PEP

Tipo de produto	Percentual de Esforço
Requisitos	20%
Análise e Projeto	30%
Codificação	35%
Testes	15%

Tabela 4 - Fatores

Fator	Valor
EF	1,00
Soma de Pesos: Peso*(Autenticar Usuário)* + Peso*(Cadastrar Usuário)*	15,00
TUCP de todo o sistema	18,36
Fator de produtividade - FPP (Análise e Projeto)	15,00

Com estas características, o tamanho total do projeto seria de 18,36 TUCP´s e para o caso de uso teríamos:

TUCP(Projeto do Caso de Uso Autenticar Usuário) = (Peso(Autenticar Usuário) / (Peso(Autenticar Usuário) + Peso(Cadastrar Usuário)) x (TUCP de todo o sistema) x (Percentual de Esforço na Análise e Projeto) = (10/(10 + 5)) x 18,36 x 0,30 = **3,62 TUCPs**

Considerando o fator de produtividade da atividade de análise e projeto como 15 teríamos um esforço de:

Esforço = (*Autenticar Usuário*) = TUCP (*Autenticar Usuário*) * EF * FPP (*Análise e Projeto*) = 3,62*1*15 = **54,03 homens-horas** para a análise e projeto do caso de uso *Autenticar Usuário*.

5. ESTUDO DE CASO

Esse estudo de caso foi realizado em uma organização de pesquisa e desenvolvimento certificada como SW-CMM nível 2 que utiliza processo RUP (Rational Unified Process) [5] [10] em projetos de pesquisa e desenvolvimento em diversas áreas de tecnologia da informação e telecomunicações, em diferentes plataformas (J2EE, J2ME e .NET).

A técnica de estimativa proposta neste artigo é aplicada em todos os projetos de software da instituição. Para esse estudo de caso, serão apresentados dois projetos com características e tamanhos diferentes. Os projetos serão denominados A e B por questões de confidenciabilidade.

O esforço estimado do projeto A foi de 28.000 homens-horas e tem como característica o desenvolvimento de um sistema para a automação de processos. O esforço estimado do projeto B foi de 2.000 homens-horas e é um sistema para cadastro e consulta de materiais que utilizou em seu desenvolvimento um framework com alto reuso e geração de código.

Para calcular o percentual de erro estimado para cada atividade utilizando a técnica proposta, foi utilizada a métrica *Symmetric Relative Error* (SER) proposta por M. Jorgensen e D. Sjobeg [11]. A fórmula é dada por:

SER = Real - Estimado / Real, se Real =< Estimado
SER = Real - Estimado / Estimado, se Real => Estimado,

onde "Real" é o esforço real do projeto e "Estimado" é o esforço estimado utilizando a técnica proposta neste artigo.

Os resultados apresentados nas Tabela 5 e Tabela 6 mostram o percentual de erro relativo entre valores estimados e realizados. Para efeitos de comparação, a Tabela 5 apresenta valores estimados com o fator de produtividade único para todos os tipos de produto, enquanto a Tabela 6 apresenta o fator de produtividade calibrado por tipo de produto de trabalho.

Na Tabela 6 os resultados apresentados mostram que o percentual de erro total em relação ao real é muito menor se comparado com a Tabela 5, onde não existiu a distribuição do fator de produtividade.

Tabela 5 - Percentual de erro sem distribuição de produtividade por tipo de produto de trabalho

Projeto	REQ	A&P	COD	TESTE	Total
A	-2,00	24,88	-49,03	29,08	-11,05
B	-47,81	-86,26	-32,96	-52,59	-46,61

Tabela 6 - Percentual de erro com distribuição de produtividade por tipo de produto de trabalho

Projeto	REQ	A&P	COD	TESTE	Total
A	-1,93	24,81	-11,76	29,08	3,24
B	-10,85	7,37	0,29	-34,61	-3,59

A calibração no fator de produtividade utilizada na Tabela 6 considera que o fator de produtividade nos projetos A e B foram

- Fator de produtividade de requisitos: entre 15 e 20 homens-horas.
- Fator de produtividade de análise e projeto: entre 20 e 25 homens-horas.
- Fator de produtividade de codificação: entre 15 e 20 homens-horas.
- Fator de produtividade de testes: entre 10 e 15 homens-horas.

Estes valores podem ser redistribuídos caso as características do projeto e da equipe influenciem em alguma atividade durante o desenvolvimento de software. Segundo a Tabela 6, os valores apresentados no projeto A para os produtos análise e projeto e teste tiveram um percentual de erro maior do que o real.

Apesar dos desvios nos percentuais de erros em alguns produtos de trabalho serem maior do que 20%, o desvio total no esforço é muito pequeno se comparado com os valores individuais para cada produto de trabalho.

Para o projeto B (ver Tabela 6), os percentuais dos erros relativos para "análise e projeto" e "codificação" foram baixos. Isso ocorreu devido à distribuição realizada nos principais produtos de trabalho, que levou em consideração que a "análise e projeto" de um sistema de consulta e cadastro exige um esforço menor. Na codificação, o esforço também foi baixo devido à geração de código através de um *framework*, que permitiu um desenvolvimento mais rápido.

Analisando a Tabela 5 e a Tabela 6, podemos verificar que a distribuição da produtividade por tipo de produto de trabalho auxiliou na precisão da estimativa de esforço. No entanto, uma base histórica para uso na calibração da produtividade é importante, pois fatores como características de projeto, plataforma utilizada e desempenho da equipe podem influenciar neste fator de produtividade.

6. CONCLUSÃO

Esse trabalho apresentou uma extensão da técnica PCU, para atender recomendações do CMMI-SW nível 2, permitindo uma visão mais detalhada das estimativas por tipo de produto de trabalho, possibilitando realizar refinamentos dessas estimativas ao longo do processo de desenvolvimento.

Essa técnica, TUCP, pode ser utilizada em projetos que utilizam casos de uso para especificação de requisitos de software.

As principais contribuições deste trabalho foram:

- A extensão do método PCU através da TUCP permitiu uma visão mais detalhada das estimativas por tipo de produto de trabalho;
- O fator de produtividade por tipo de produto de trabalho gerou menor erro na estimativa total do projeto e na estimativa de cada produto de trabalho, possibilitando assim um planejamento e acompanhamento mais efetivos;
- A abordagem proposta possibilita fazer refinamentos nas estimativas nas várias fases do processo de desenvolvimento;
- Um fornecimento de uma técnica de estimativa aderente às práticas dos modelos CMMI-SW, como também do SW-CMM [6][7].

Como conclusões mais importantes deste trabalho, pode-se citar:

- Uma base histórica com as estimativas da organização deve ser implementada (como recomenda a literatura de métricas de software), afim de que sirvam de fundamentação nas calibrações, e nos fatores de produtividade para projetos futuros;
- A elaboração do caso de uso deve ser descrita em um nível de detalhamento adequado, para que a estimativa baseada em PCU seja eficaz;

- A inexistência de padrões aceitos universalmente para a construção de casos de uso dificulta a comparação entre projetos de diferentes organizações. Não há como garantir que os TUCP´s estarão medindo a mesma coisa, se os critérios utilizados para construir os casos de uso forem muito diversificados.

Referências Bibliográficas

[1] Anda, B. *Comparing Effort estimates Based on Use Case Points with Expert Estimates*, 4th International Conference, Toronto, Canada, October 1-5, 2001, LNCS 218, 2001.

[2] *CMMI[SM] for Systems Engineering*/Software Engineering, Version 1.1, Staged Representation (CMMI-SE/SW, V1.1, Staged). Disponível em: http://www.sei.cmu.edu/publications/documents/02.reports/02tr002.html. Acesso em 27/04/2004.

[3] *ISO/IEC 15504: Information Technology – Process Assessment*, Part 1 to Part 5, 2003.

[4] Karner, G. *Metrics for Objectory*. Diploma thesis, University of Linköping, Sweden. No. LiTH-IDA-Ex-9344:21. December 1993.

[5] Kruchten, Philippe. *The Rational Unified Process - An Introduction*. 2nd Ed. New Jersey: Addison-Wesley, 2000.

[6] Paulk, M. C. et al. *Capability Maturity Model for Software*, Version 1.1, SEI-CMU-93-TR-24, Software Engineering Institute, 1993.

[7] Paulk, M., Weber, C. *The Capability Maturity Model: Guidelines for Improving the Software Process*, 17th ed. Addison-Wesley. ISBN 0-201-54664-7, 2003.

[8] Schneider, G., Winters, J. *Applying Use Case: A Practical Guide*. 2nd ed. Addison-Wesley, 2001. ISBN 0-201-70853-1.

[9] Symons, C.R. *Software Sizing and Estimating*, MKII FPA. Jonhn Wiley and Sons, 1991.

[10] Rational Software Corporation, *Rational Unified Process*, version 2001.3, CD-ROM, Rational Software, Cupertino, Calif.:2001.

[11] Ribu, K. *Estimating Object-Oriented Software Projects with Use Cases*. Masters Thesis, University of Oslo. November 2001.

Projetando um Serviço de Descoberta de Canais para TV Digital

Juliana R. B. Diniz Barros [1,2], Adriana R. Silva[2], Roberto S. M. Barros[2], Carlos A. G. Ferraz[2], Nelson S. Rosa[2]

[1]Faculdade Integrada do Recife, Coordenação de Pesquisa e Pós-graduação,
Av. Abdias de Carvalho, 1678 – 50720-635 – Recife-PE – Brasil

[2]Universidade Federal de Pernambuco, Centro de Informática
Caixa Postal 7851 – 50732-970 – Recife – PE – Brasil

juliana@fir.br, {ars, roberto, cagf, nsr}@cin.ufpe.br

Abstract

Nowadays, there are several proposals in the context of Digital TV in Brazil and around the world. Applications to be used in this scenario are being developed and a considerable number of channels will be available to the Digital TV users soon. Therefore, the existence of a channel look up service is going to become important, because memorizing a large the number of channels is not going to be realistic, especially when the mobility of the users in vehicles is considered. This paper *describes the design of a Digital TV registration and look up service using XML.*

Keywords: Digital TV, XML, Registration, Look up, Middleware

Resumo

Diversas propostas referentes a TV Digital estão sendo desenvolvidas no âmbito mundial. Entretanto muitos paises como o Brasil, por exemplo, ainda não sabem que sistema será adotado, muito embora, as pesquisas e interesses pelas operadoras de TV e desenvolvedores já sejam bem significativas. Aplicações para serem utilizadas dentro do cenário da TV digital vêm sendo projetadas, e possivelmente um número muito extenso de canais será disponibilizado aos seus usuários. Dessa forma, é muito importante a criação de um serviço de localização de canais, pois ficará inviável para os usuários memorizarem um número muito grande de canais, principalmente quando se considera que este usuário se desloca com a sua TV em um veículo e pretende continuar assistindo a sua programação durante o seu deslocamento. Sendo assim, a proposta deste artigo é desenvolver um serviço de registro e seleção de canais para a TV Digital, usando para isso a linguagem XML.

Palavras chaves: TV Digital, XML, Registro, Seleção, *Middleware*

1. Introdução

Nos últimos 50 anos, as emissoras usaram sinais analógicos como um meio de transmissão de TV. Durante este período, foi vista a transição da TV preto e branco para a TV a cores. Esta migração fez com que as pessoas trocassem seus aparelhos de televisão, e as emissoras adquirissem novos transmissores e equipamentos de produção. Esta mudança trouxe benefícios para todos.

Atualmente, a industria está se preparando para mais uma mudança revolucionária: a migração da TV convencional para uma nova tecnologia digital. As operadoras de televisão já estão se preparando para a adaptação das suas redes existentes como também a introdução de avançadas plataformas digitais para que possibilite novas oportunidades para consumidores e fornecedores de conteúdos.

O projeto DVB-Digital Video Broadcasting abrange mais de 200 organizações em mais de 30 países, tendo como objetivo criar as condições técnicas necessárias para o desenvolvimento e a implantação da televisão digital na Europa. De acordo com Pereira [10], as normas criadas pelo projeto DVB consideram a difusão televisiva em vários meios de transmissão. Podem ser citados os seguintes meios, dentre outros:

- DVB-S - especifica um sistema para difusão de televisão digital via satélite;
- DVB-C - especifica um sistema para difusão via cabo capaz de se adaptar às características de qualquer rede de cabo;
- DVB-T - especifica um sistema para difusão terrestre (norma ETSI EN 300 744 de 1997);
- DVB-MC/S - especifica um sistema para difusão por microondas.

Até o presente momento, os serviços de rede de difusão DVB-T e os serviços de dados sobre redes móveis da terceira geração, baseados nas tecnologias UMTS/GPRS [16] operam independentemente. Os serviços multimídia para computação móvel têm se tornado populares, embora ainda exista um número bastante limitado desses serviços sobre redes digitais de comunicação móveis. As redes de difusão que oferecem serviços digitais, tais como TV, rádio e dados já estão atuando há um certo tempo.

O projeto europeu CISMUNDUS (Convergence of IP-based Services for Mobile Users) [2] tem um papel fundamental para o desenvolvimento desta tecnologia pois vem desenvolvendo conceitos inovadores de serviços de publicação de conteúdo multimídia aos usuários móveis utilizando a rede de difusão DVB-T e a rede de telecomunicações UMTS/GPRS. A proposta do projeto CISMUNDUS é ajudar na criação de um novo ambiente de comunicação que combine diversas ferramentas com tal objetivo para indivíduos ou grupos que estão em deslocamento, ou seja, usuários móveis. O projeto CISMUNDUS em conjunto com outros projetos como SAVANT [14], CONFLUENT [3] e SAMBITS [11] motivaram o surgimento de um projeto integrado maior denominado INSTINCT [5]. O objetivo do projeto INSTINCT é agregar operações de redes de difusão DVB, tais como negócios, conteúdo, serviços, aplicações e infra-estrutura de redes, com as redes celulares da segunda e terceira geração, proporcionando acesso aos usuários e serviços de multimídia avançados através de dispositivos pessoais.

De acordo com Cosmas et. al. [4] e Launau et. al. [6], a chave para o sucesso das comunicações móveis do futuro está exatamente nos serviços atrativos que devem ser oferecidos aos usuários finais, uma vez que a motivação para os negócios nesta área está no aumento da disponibilidade de conteúdo e serviços através da expansão do acesso aos serviços, devido a infra-estrutura de rede e terminais de acesso.

O baixo custo será a chave para o sucesso desses novos serviços oferecidos aos clientes que estão condicionados ao acesso barato aos serviços de Internet e telefonia convencional e ao acesso livre às redes de difusão. Os sistemas celulares da segunda (2G) e terceira (3G) geração devem ser capazes de oferecer diversos serviços interativos e personalizados aos clientes que desejarem acessar páginas de interesse pessoal na Internet. Entretanto, sabe-se que as técnicas de telefonia celular não proporcionam mecanismos de *download* de grande volume de dados a um baixo custo. Os sistemas de difusão, por sua vez, podem entregar dados em grande quantidade a um custo relativamente baixo, porém apresentam pouca ou nenhuma interatividade ou personalização de conteúdo, uma vez que os dados são transmitidos de uma vez a todos os receptores de uma área de cobertura.

Dentro do contexto da TV Digital, a interação com o usuário deve ser desenvolvida utilizando-se dispositivos que lhes são familiares, por exemplo, telefones celulares, onde o receptor de difusão não está necessariamente conectado ao dispositivo de interação com o usuário. Atualmente os pontos críticos para o desenvolvimento de dispositivos de difusão compatíveis com telefones móveis residem no consumo de energia e na miniaturização do terminal *front-end.* É importante salientar que para a utilização e interoperabilidade entre os dispositivos será requerido o desenvolvimento e a especificação de um *middleware*[1] que habilite o mesmo dispositivo a operar em várias situações, principalmente quando se fala em dispositivos móveis. Entre outras funções, o *middleware* deverá possuir serviços de descobertas, que poderá conter mecanismos disponíveis em DVB e *Web Services*, serviços de descoberta e configuração dinâmica dentre outros. É essencial que este *middleware* trabalhe com uma grande diversidade de terminais e dispositivos. Alguns cenários motivados pelo projeto INSTINCT podem ser observados na Figura 1.

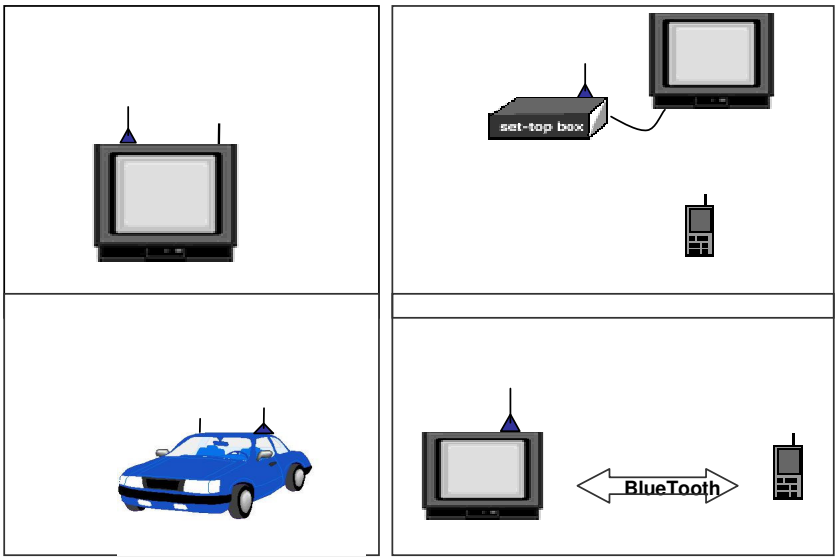


Figura 1: Cenários motivados pelo projeto INSTINCT

Este artigo aborda a descrição de um serviço de descoberta e seleção de canais para a TV Digital utilizando a linguagem de marcação XML. Esse serviço é um serviço distribuído e sensível a contexto que pode ser adaptado a um *middleware* que esteja habilitado a trabalhar com uma rede cujas estações clientes podem se mover sem, no entanto, perderem a conexão ao serviço.

Este serviço de seleção de canais e descoberta de serviços é considerado como uma característica essencial para os futuros *middleware* que estão sendo implementados para dar suporte à TV Digital. Além desses serviços também considera-se importante serviços de configuração e conectividade dinâmica, interação com novas entidades (gerenciamento de aplicações distribuídas com muitos dispositivos conectados), dentre outros, que não são tratados neste artigo.

O restante deste artigo abrange 5 seções. A Seção 2 trata dos trabalhos relacionados. A terceira Seção traz os fndamentos, concentrando o desenvolvimento deste trabalho no nível de *middleware* e serviços para as redes convergentes DVB-T e telefonia celular. A Seção 4 realiza a descrição do serviço de registro e localização de canais para a TV Digital, enquanto que a quinta Seção trás o protótipo desenvolvido. A Seção 6 apresenta as conclusões e as propostas de trabalhos futuros.

2. Trabalhos Relacionados

Serviços similares ao desenvolvido por este artigo podem ser encontrados em *middleware* baseado na arquitetura CORBA [9] de objetos distribuídos e na arquitetura de *Web Services* [15] no ambiente Internet. Neste último, o serviço de descoberta é o UDDI (*Universal Description Discovery and Integration*).

O UDDI provê um mecanismo para registro e localização de *Web Services* na Internet. Trata-se de um diretório que contém vários *Web Services* registrados, onde um nome é associado a um serviço. Através dele, empresas registram seus serviços e podem interagir com outras empresas interessadas naquele tipo de serviço. O próprio UDDI é um *Web Service* baseado em XML e SOAP (*Simple Object Access Protocol*).

Um outro exemplo de serviços de páginas amarelas previamente citado é o serviço de nomes (*naming)* utilizado pela arquitetura CORBA. Através deste serviço, qualquer processo cliente pode fazer acesso a um objeto remoto conectando-se inicialmente ao serviço de nomes e passando o nome do serviço pelo qual ele procura. O serviço de nomes, por sua vez, retorna o IOR (*Identify Object Reference)*, ou seja, uma cadeia numérica referente a identificação do objeto dentro do sistema e com essa referência em mãos, qualquer atividade pode ser solicitada ao objeto pelo seu cliente. Parte-se do princípio de que é bem mais fácil a manipulação com nomes que com cadeias numéricas.

Uma aplicabilidade deste serviço pode ser observada através de um cenário utilizando um automóvel. Pode-se imaginar um veículo se deslocando em uma rodovia, e os seus passageiros estão assistindo a TV Digital. Para a TV analógica, os televisores possuem antenas receptoras que através de uma seleção prévia de canais faz com que o usuário

tenha acesso aos mesmos. Por exemplo, sabe-se que na cidade X o canal 2 corresponde a rede Y. Cada canal possui a sua faixa de freqüência previamente estabelecida em cada região cabendo ao telespectador sintonizar no canal desejado para assisti-lo.

Em se tratando de estações clientes receptoras móveis, o número de canais disponíveis seria maior e o usuário não teria condições de memorizar ou sintonizar manualmente toda a lista de canais disponíveis. Dessa forma, a modelagem e o desenvolvimento de um serviço de páginas amarelas ou de descoberta de canais seria importante para automatizar o processo. Assim, quando um usuário em um veículo entra numa região, a estação receptora conecta-se com o serviço de páginas amarelas de canais solicitando a lista de canais, podendo inclusive diretamente fornecer o nome do canal desejado e tendo a sintonia automática.

Seguindo esta abordagem, o serviço de seleção de canais visa facilitar a vida do usuário dos recursos disponíveis para a TV Digital, uma vez que o mesmo só precisará conhecer o nome do canal que ele procura porque o serviço tratará de encontrá-lo naquela região. Outras atividades avançadas também podem ser desempenhadas por este serviço, como, por exemplo, listar canais de uma determinada categoria, listar programas de uma categoria, listar a programação de um canal, dentre outras. Além disso é importante ressaltar que tal serviço proporcionará o seu uso dentro do cenário onde os usuários se movem a todo instante e este fato não poderá diferenciar do serviço oferecido a esses usuários dos serviços que seriam oferecidos caso não houvesse a mobilidade. A descrição de todas as operações para o desenvolvimento desse serviço neste trabalho foi desenvolvida em XML. A utilização desta linguagem se dá pela portabilidade de plataformas.

3. Fundamentos

De acordo com o projeto INSTINCT, existem cinco domínios no modelo de referência para representar a cadeia de valores para os sistemas convergentes no contexto da TV Digital. São eles: fornecedores de conteúdo, provedores de serviços, operadores de redes, provedores de aplicação e usuários finais. Esses domínios são agrupados verticalmente em três correntes, a saber:

- Conteúdo, serviços e aplicações para sistemas híbridos de redes de telefonia celular e difusão;
- Configuração de redes suportando tráfego sobre redes híbridas;
- Dispositivos de usuários, para que estes possam ter acesso aos serviços e conteúdo usando mecanismos disponíveis nas redes híbridas.

Para as atividades relacionadas a conteúdo, serviços e aplicações alguns requisitos podem ser observados. Por exemplo, no nível de criação é necessário um conjunto de ferramentas para criar meios para serviços usando formatos já existentes como MPEG2 e MPEG4 para a parte gráfica. Pode-se utilizar XML e XSLT para conteúdo Internet.

A divisão entre domínios e correntes, conforme tratada pelo projeto INSTINCT, pode ser observada na Figura 2. A corrente referente a dispositivos de usuário final subdivide-se em dois tipos de atividades identificadas como chave de acordo com o projeto: terminais móveis habilitados para redes de difusão terrestre e camadas de *middleware* e aplicação combinando os ambientes baseados em difusão e compatíveis com a mobilidade.

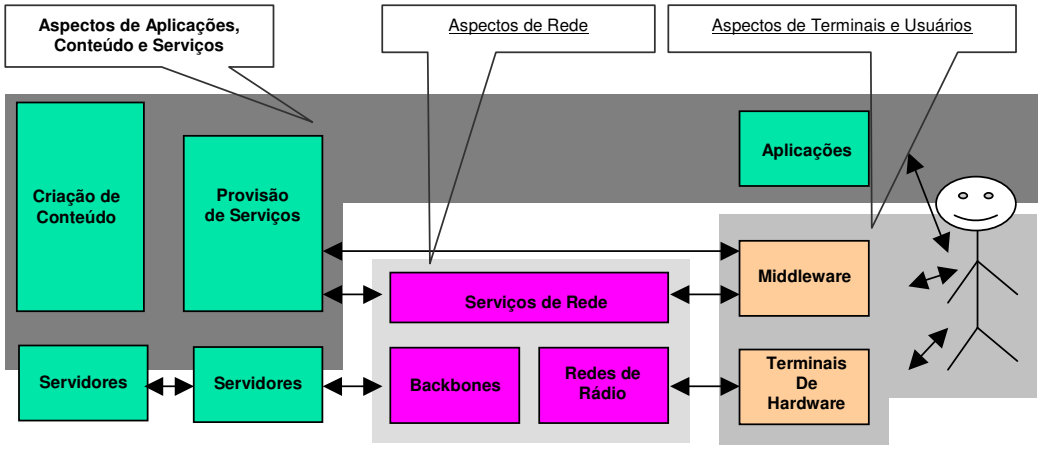


Figura 2: Modelo de referência do projeto INSTINCT

Um requisito do usuário é a criação de condições de acesso transparentes aos serviços não considerando a proposta do equipamento do usuário final, nem se o usuário é integrado ou distribuído. Tecnicamente isto consiste em identificar e validar as extensões requeridas para os padrões existentes para os diversos níveis de API (*Application Program Interface)* como por exemplo, MHP[8] para dispositivos DVB e MIDP [13] para os dispositivos celulares. As extensões incluem suporte a interoperabilidade de dispositivos distribuídos para cenários relevantes e a adaptação das interfaces dos usuários para um perfil amigável no contexto dos dispositivos distribuídos.

De acordo com o modelo de referência do projeto INSTINCT, este trabalho concentra-se no domínio dos provedores de aplicações e serviços, mais especificamente no componente *midlleware* e na intermediação de serviços entre o usuário final e os operadores de redes. pretende-se tratar o serviço de seleção e registro de canais que será prestado sobre um *middleware* compatível com a mobilidade e será requisitado pelos usuários finais**.**
Um *middleware* é uma camada de software que abstrai a heterogeneidade e a complexidade dos sistemas distribuídos tornando detalhes de baixo nível transparentes aos desenvolvedores e usuários da aplicação.

A tarefa do *middleware*, no cenário da TV digital, é prover a comunicação entre os dispositivos distribuídos proporcionando que as aplicações utilizem os serviços disponíveis e gerenciados, pelo menos, enquanto viabilizam que os usuários façam uso de novas experiências de dispositivos, serviços e sistemas [5].

Quando desenvolvem aplicações distribuídas, os projetistas não se preocupam com problemas relativos à heterogeneidade, escalabilidade, dentre outros. A proposta do *middleware* é exatamente tornar estes detalhes transparentes a nível de programação e desenvolvimento. Isto tira tal responsabilidade das mãos do projetista, fazendo com que ele possa de fato se concentrar naquilo que ele realmente está interessado.

Para atender os requisitos impostos pela mobilidade, o *middleware* deve ser projetado com o suporte apropriado ao desenvolvimento de aplicações móveis. A maioria das propostas satisfazem às necessidades de aplicações particulares e não são suficientemente generalizadas a ponto de serem re-usáveis. Em geral, os modelos de *middleware* existentes (por exemplo: orientado a mensagens, orientado a transação, orientado a objetos) foram construídos aderindo à proposta da *caixa preta*, isto é, a existência de muitos componentes distribuídos é escondida dos usuários e desenvolvedores e o sistema aparece como um ambiente único integrado. Tal abordagem tem bastante sucesso quando se trata de sistemas distribuídos construídos sobre redes fixas, mas não é adequada quando se trata de ambientes de redes móveis, onde as estações podem mover-se ao longo da rede. A indisponibilidade, a baixa largura de banda, a falta de acoplamento e as atualizações dinâmicas (por exemplo, de localização, configuração, disponibilidade de serviços) são características normais em tais ambientes e não exceções. Os sistemas móveis precisam detectar e adaptar-se às mudanças drásticas que acontecem no ambiente. Desta forma, é necessário adaptar os princípios das soluções de *middleware* que se adequam à transparência, em favor do novo paradigma de *awareness* (ciência), que permite ao projetista inspecionar o contexto e adaptar o comportamento do *middleware* de acordo com as mudanças de contexto.

4. O Serviço de Descoberta e Seleção de Canais

O serviço de seleção de canais tem como objetivo prover aos usuários da TV Digital formas de seleção de canais otimizadas ou especializadas como, por exemplo, listar todos os canais que transmitem filmes ou a programação de um canal pelo horário desejado. Também é preocupação do serviço de seleção de canais, assegurar a transmissão, não permitindo a perda do sinal, visto que será possível a utilização de dispositivos móveis para captar o sinal da TV Digital. O serviço preocupa-se em buscar o transmissor da região onde se encontra, permitindo um serviço transparente para o usuário caso ele esteja se deslocando de uma região para outra.

O cenário de mobilidade para a TV digital pode ser observado na Figura 3.
Neste cenário, o servidor é fixo para uma macro-região e os clientes movem-se, pois são representados por veículos. Uma estação transmissora de um canal faz o papel de servidor e atende a uma macro-região, pois o alcance de uma antena transmissora de TV atende a um conjunto de células (regiões menores). Dentro de uma macro-região existem diversas células menores. Quando um receptor (estação móvel) passa de uma célula para outra, ele passa a ser atendido pelo serviço de localização e registro de canais daquela outra célula. Quando este receptor passa de uma macro-região para outra, o mesmo deverá se comunicar com o serviço da sua nova célula que, por sua vez, já tem registrado os servidores de canais da nova macro-região.

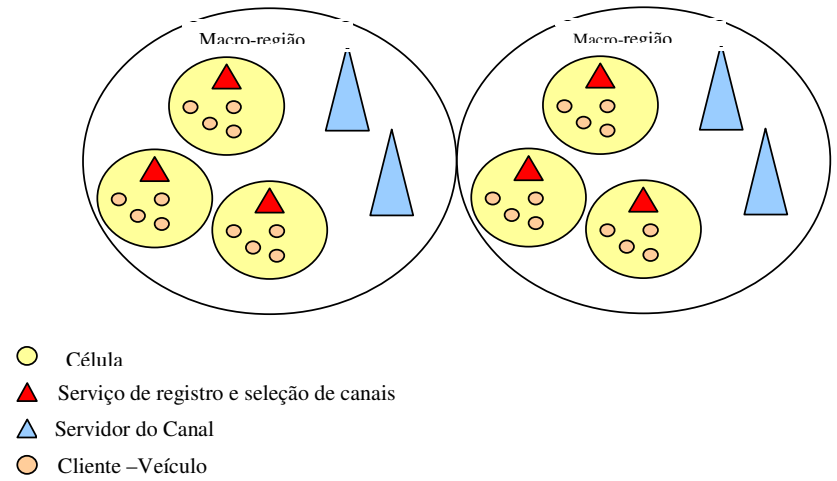


Figura 3: Cenário da Mobilidade para TV Digital

Este serviço possui dois conjuntos de operações fundamentais e complementares: a publicação do serviço e a solicitação do serviço.

Nas atividades de publicação, o serviço funciona da seguinte maneira: a transmissora do canal cadastra o mesmo juntamente com a programação no serviço de registro de canais local que atende às estações clientes receptoras que se encontram dentro daquela célula.

A visão geral dos componentes de software do sistema pode ser observada na Figura 4.

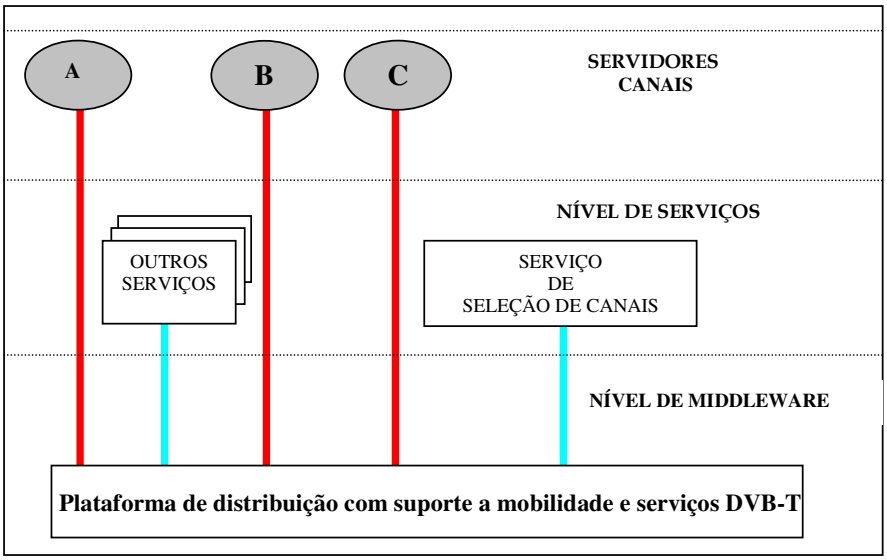


Figura 4: Visão geral dos Componentes de Software do Sistema

Na Figura 4, nota-se a existência de três níveis onde o mais inferior refere-se ao *middleware* para suportar tal serviço e o mais superior refere-se aos servidores de canais da rede de difusão terrestre, ou seja, são as emissoras de televisão que estão distribuídas ao longo de uma determinada área de cobertura que será atendida pelo serviço de seleção de canais local. O nível intermediário está exatamente ocupado por este serviço que localiza os canais a partir de alguns parâmetros que foram fornecidos pelos usuários no momento da requisição. Outros serviços específicos também poderão ser encontrados neste nível.

O serviço de seleção de canais baseia-se na linguagem XML. Os canais e os seus respectivos programas se encontram em um arquivo de extensão *xml*, que é utilizado como repositório de dados dos canais e seus programas. A Figura 5 exibe um fragmento de um documento XML. Neste exemplo, observa-se a definição de um canal denominado

"TVU", cuja categoria é científica, e possui um jornal científico como programa que acontece das 10 às 11 horas, também com categoria científica.

```
<channel channel_id="TVU">
        <channel_number>1</channel_number>
        <channel_description>Canal com programas de cunho cientifico</channel_description>
        <channel_category>Cientifico</channel_category>
        <status>String</status>
        <program program_id="U_NOTICIAS">
            <program_censura>LIVRE</program_censura>
            <program_description>Jornal de noticias de trabalhos desenvolvidos no ramo da
informatica</program_description>
            <program_category>Cientifico</program_category>
            <program_hour>10:00</program_hour>
            <program_duration>1hora</program_duration>
        </program>
    </channel>
```

Figura 5: Parte do documento XML – Dados do canal e programa armazenado

Nas atividades de solicitação, o usuário do serviço tem as seguintes formas de fazer uma requisição ao serviço de localização de canais:

- Através do nome do canal, onde será exibido o programa que está sendo apresentado no momento naquele canal;
- Através do nome do canal solicitando a programação deste, onde será listada toda a programação desse canal;
- Através da categoria de canal, onde serão listados todos os canais dessa categoria;
- Através da categoria de programa, onde serão listados todos os programas dessa categoria;
- Através do horário de programa, onde serão listados todos os programas exibidos nesse horário.

4.1 Funções de Solicitação do Cliente

Inicialmente, toda descoberta de canais e buscas personalizadas só podem ser realizadas após a conexão do cliente com o serviço de registro e seleção da região. Esta operação acontece de forma transparente sem que o cliente tome conhecimento, pois a conexão com o serviço é espontânea.

Uma vez que o cliente está conectado ao serviço de nomes que atende aos canais de uma região, todas as funções de busca e seleção de canais desta região poderão ser desempenhadas para prover maior interação do mesmo com os serviços DVB-T através da rede 2G e 3G de telefonia celular.

A primeira função de busca personalizada do cliente retorna uma lista de canais de uma determinada categoria, pois a mesma habilita que o cliente tenha disponível em um mecanismo de identificação de todos os canais da categoria desejada. Caso não exista nenhum canal com aquela categoria, a mensagem de retorno conterá um elemento vazio. Caso não seja passado nenhum argumento de busca, não será retornado nenhum conjunto de resultados. Caso o nome da categoria cujos canais devem ser buscados não seja passado, todos os canais serão retornados na lista de canais. A descrição em XML desta operação pode ser observada na Figura 6. Neste documento faz-se a requisição por canais da categoria "NOTÍCIAS".

```
<?xml version="1.0" encoding="UTF-8"?>
<find_category xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Digital_TV_API.xsd">
<channel_category>NOTICIAS</channel_category>
</find_category>
```

Figura 6: Parte do documento XML - Busca canais de uma categoria específica

Uma vez que o cliente obteve uma lista de canais de acordo com a categoria desejada, o mesmo poderá selecionar um dos canais para assisti-lo. Para isso ele usará a função de seleção de canal pelo nome. Esta operação retorna o canal que foi procurado pelo nome, caso a solicitação esteja correta. Caso o nome do canal não seja passado, será retornada uma mensagem avisando ao usuário que ele deve refazer a busca. A requisição em XML da operação pode ser observada na Figura 7. Neste caso um canal de nome TVU está sendo procurado, e sua imagem deverá ser transmitida.

```
<?xml version="1.0" encoding="UTF-8"?>
<find_program xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation=" Digital_TV_API.xsd">
    <program_category>DOCUMENTARIO</program_category>
</find_program>
```

Figura 7: Parte do documento XML - Busca um canal pelo nome

Também é possível procurar os programas de uma categoria que serão exibidos nas próximas 4 horas em todos os canais de uma determinada região. A descrição XML desta função é ilustrada na Figura 8. Nota-se que se deseja fazer uma busca por programas da categoria "DOCUMENTÁRIO" que acontecerão nas próximas 4 horas em todos os canais da célula atual.

Também é permitido fazer uma busca mais elaborada e procurar programas por categoria e horário. Esta operação retorna uma lista de programas de uma determinada categoria que serão exibidos em um determinado horário em todos os canais. A categoria e o horário são utilizados como chaves na busca, caso a solicitação esteja correta. Essa descrição é ilustrada na Figura 9. Neste caso, procura-se por programas documentários que são exibidos às 8 horas.

```
<?xml version="1.0" encoding="UTF-8"?>
<find_channel xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Digital_TV_API.xsd">
<channel>TVU</channel>
</find_channel>
```

Figura 8: Parte do documento XML - Busca programas de uma categoria em todos os canais

```
<?xml version="1.0" encoding="UTF-8"?>
<find_programhours xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Digital_TV_API.xsd">
    <program_category>INFANTIL</program_category>
    <program_hour>08:00</program_hour>
</find_programhours>
```

Figura 9: Parte do documento XML - Busca programas de uma categoria em todos os canais por horário

4.2 Funções de registro de canais

Para que um canal possa ser acessado por algum receptor é necessário que ele esteja registrado no serviço de registro e seleção de canais da região. Uma vez que o canal foi registrado, o mesmo tornar-se-á disponível para ser procurado por algumas das funções de busca personalizadas. A função de registro de canais em um serviço de canais pode ser observada na Figura 10. Esta requisição retorna uma mensagem contendo as novas informações registradas sobre aquele canal no serviço, sinalizando que a atualização aconteceu com sucesso, e gerando um identificador para o novo canal automaticamente.

```
<?xml version="1.0" encoding="UTF-8"?>
<add_channel xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Digital_TV_API.xsd" channel_id="TVCIN">
    <channel_number>10</channel_number>
    <channel_description>Canal com as noticias do CIN</channel_description>
    <channel_category>Ciencia</channel_category>
    <status/>
</add_channel>
```

Figura 10: Parte do documento XML - Registra um canal no serviço

5. Protótipo

Para proporcionar a validação do serviço desenvolvido por este trabalho foi implementado um protótipo. O protótipo utilizou o modelo de *middleware* baseado na arquitetura de *Web Services.* Como citado previamente, esta arquitetura apresenta um serviço denominado UDDI que provê um mecanismo para registro e localização de *Web Services* na Internet.

Inicialmente pensou-se em substituir o modelo de descoberta de serviços UDDI pelo serviço de registro e localização de canais, utilizando então o modelo de *Web Services* provido por alguma das ferramentas que o implementa e oferece código aberto. Dessa forma o UDDI poderia ser trocado pelo serviço de registro e seleção de canais da TV Digital. Entretanto, o modelo *Web Services* não proporciona tal flexibilidade, de modo que o mecanismo de descoberta de serviços é sempre feito pelo UDDI. O produto que foi avaliado para propor esta solução foi o Java *Web Services* [12]. Porém o modelo de *Web Services* tem o padrão de utilizar o UDDI não possibilitando o uso de outro serviço de registro. Dessa forma a comunicação teve que ser implementada utilizando mecanismos de *sockets,* utilizando alguns fundamentos do modelo *Web Services*, como por exemplo, os protocolos HTTP e SOAP.

Como citado previamente, também seguindo os fundamentos do modelo *Web Services*, utilizou-se a linguagem XML para a definição das operações do serviço por esta ser independente de plataformas.

Levando em consideração a falta de flexibilidade imposta pelo UDDI e pelo modelo *Web Services*, foi definida uma comunicação através da implementação do protocolo SOAP. Este protocolo é encapsulado no HTTP, permitindo a sua passagem pelas barreiras impostas pelo *FireWalls*. Ressalta-se ainda o fato de aplicações sob redes de telefonia móvel baseadas em GSM (*Group Special Mobile*) utilizarem as tecnologias baseadas no protocolo HTTP. Pode-se citar como exemplo, o protocolo WAP[18] que possui um WAP *Gateway,* que é basicamente um software que realiza uma conexão entre um cliente (dispositivo móvel) e um servidor (HTTP *Server*, HTTP *Proxy*, etc). O envelope SOAP contém uma mensagem XML com a requisição da operação que foi solicitada e passa através de um *socket* encapsulada num pacote HTTP.

O protótipo foi implementado usando a linguagem JAVA. A comunicação entre o cliente e o serviço foi possível através do mecanismo de *sockets* proporcionado pela biblioteca *java.net.* Também foi utilizada a biblioteca JDOM possibilitando a busca em um arquivo XML referenciado como a base de dados e para a geração de um envelope SOAP de resposta à solicitação do cliente.

Os dados sobre todos os canais estão armazenados em um arquivo XML na base de dados do serviço de registro e seleção de canais. As informações sobre cada canal são inseridas neste arquivo no momento do registro do canal no serviço.

Na Figura 11, observa-se a seqüência de operações de uma solicitação ao serviço. Inicialmente, o cliente requisita uma operação de seleção de canais. Após a submissão, um envelope SOAP é criado contendo uma mensagem XML referente à requisição e os dados de entrada para possibilitar a busca. Esse envelope é transmitido através do *socket* para o serviço de registro e localização de canais e ao receber esse envelope SOAP, o serviço lê o mesmo utilizando a biblioteca JDOM para encontrar o método e o seu valor. Com o método e o valor em mãos, o serviço busca no arquivo XML da base de dados, utilizando também o JDOM, uma resposta para a solicitação. Caso este dado seja encontrado, ele é empacotado em um outro envelope SOAP de resposta e devolvido ao cliente via *socket*. O cliente ao receber o envelope SOAP, lê o envelope e extrai a resposta, imprimindo-a na interface do usuário.

Em resumo, as seguintes tecnologias no modelo do serviço de registro e seleção de canais são usadas no protótipo para validação: O protocolo SOAP, utilizado para envio da solicitação ao serviço de registro, seleção de canais e retorno da mesma para a estação cliente; O protocolo HTTP, por sua vez é um protocolo de aplicação utilizado para troca de dados encapsulados no envelope SOAP; Os *sockets* são as estratégias de comunicação utilizadas para troca de mensagens entre a estação cliente e o serviço de canais; e por fim, da linguagem XML, utiliza-se o XML *Schema* para validar o arquivo XML da base de dados que por sua vez é consultado usando a biblioteca JDOM. Esta biblioteca também é usada para geração do envelope SOAP de resposta.

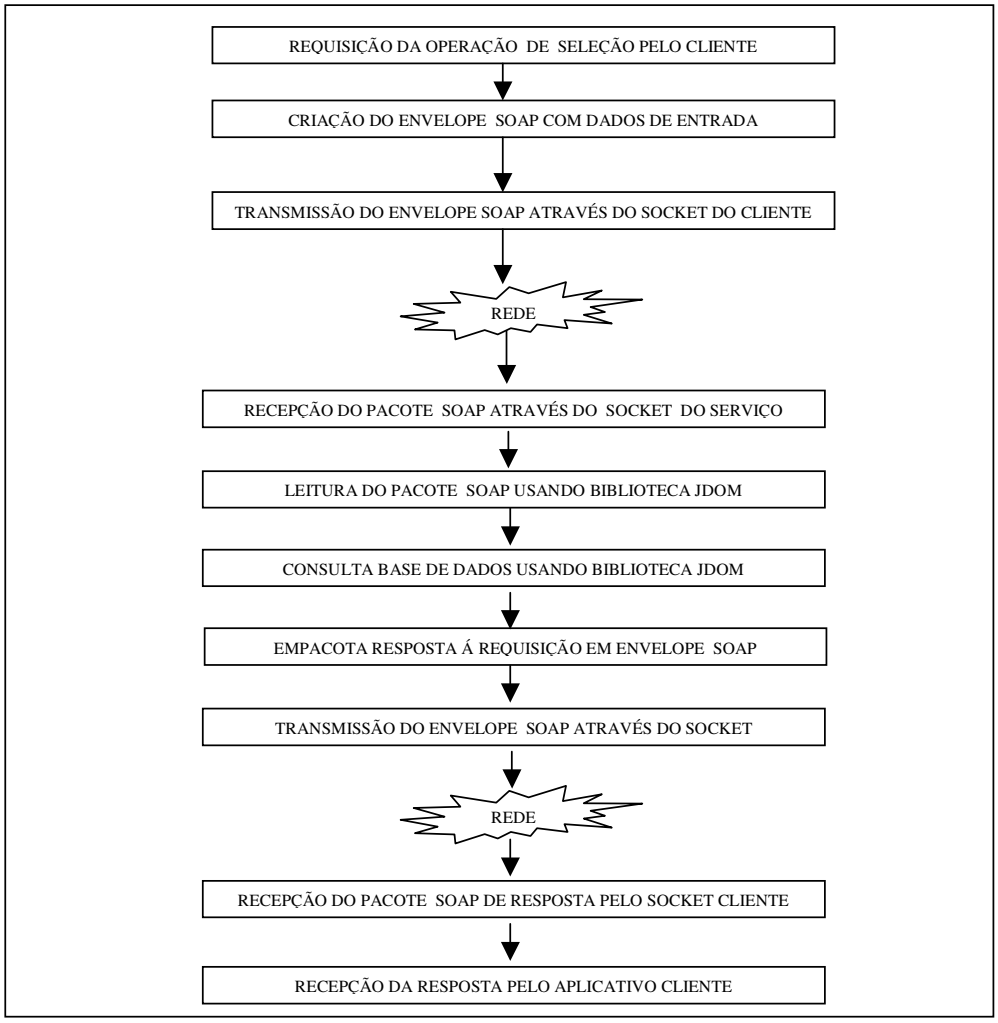


Figura 11: Seqüência de Operações

6. Conclusões

O presente artigo tratou da motivação e dos conceitos básicos da TV Digital, citando os diversos projetos em desenvolvimento em todo mundo, ressaltando o projeto INSTINCT, projeto europeu do qual o Brasil faz parte.

Dentro da proposta do projeto INSTINCT, o foco do artigo está no nível de *middleware* e serviços. Foi desenvolvido um serviço e descoberta que realiza o registro e a seleção de canais para a TV Digital que pôde, por fim, ser testado e validado.

Este serviço apresenta uma funcionalidade especial para os usuários móveis da TV digital, podendo ser adaptado a um *middleware* que atenda aos requisitos de mobilidade e seja adequado ao cenário da TV digital, considerando que o mesmo resolva problemas de desconexão abrupta, variação da largura de banda, dentre outros aspectos presentes em ambientes móveis. Outros serviços também podem ser desenvolvidos e adaptados ao *middleware* provendo um ambiente completo.

Uma vez que projetos que envolvam infra-estrutura, serviços e aplicações para a TV digital estão em desenvolvimento em vários países do mundo, a definição e a implementação de um serviço com tais categorias tem uma importante contribuição.

A linguagem XML tem um importante papel no modelo de serviços utilizados para dar suporte às aplicações da TV digital e também na troca de conteúdo entre as entidades envolvidas na comunicação. O serviço aqui especificado, juntamente com outros serviços essenciais, pode vir a contribuir para o desenvolvimento de um *middleware* específico para TV Digital.

O desenvolvimento deste serviço foi validado através de um protótipo que utilizou mecanismos de comunicação de baixo nível, fim a fim. O mesmo serviço poderia ser implementado como uma camada superior em

um *middleware* baseado em espaço de tuplas, como, por exemplo, LIME[7] ou JavaSpaces[17], adequado a mobilidade, sendo esta uma proposta de trabalho futuro.

Referências

[1] Bernstain P. A. Midleware: A Model for Distributed System Services. Communications of the ACM, (39) 2,pp 87-98. Feb 1996

[2] CISMUNDUS Project. Disponível em: http://www.brunel.ac.uk/project/Cismundus. Acesso em 15/07/2003.

[3] CONFLUENT Project. Disponível em: http://www.brunel.ac.uk/project/confluent/home.html. Acesso em 21/08/2003.

[4]Cosmas J., Itegaki T., Cruickshank L., Zheng L., Krishnapillai K., Lucas A., Elgohari L. "System Concept of a Novel Converging DVB -T and UMTS Mobile System" IEE & IEEE London Communications Symposium, University College London, pp 97-100, 9 th – 10 th. September 2002.

[5]Dosch C. , Owens T. "The Pan European Integrated INSTINCT Project - Making the Convergence of Digital Broadcasting and Mobile Communication a Reality". DVB World 2004 International Conference. Dublin, Ireland, March 2004.

[6]Launay E., Sicre J-L. , Gegout A., Belin P., "Telco and Broadcast Mobile Networks Interworking: Analysis of Service Opportunities, Technical Trends, Market and Regulations" World Telecommunication Congress, Dysneyland Paris, (WTC 2002) 23-27/09/2002.

[7]Lime Team. LIME Web page. Disponível em : http://lime.sourceforge.net. Acesso em 10/02/2004

[8] MHP Multimídia Home Platform Disponível em: http://www.mhp.org/what_is_mhp/index.html. Acesso em 20/08/2003

[9]OBJECT MANAGEMENT GROUP. "The Common Object Request Broker: Architecture and Specification Revision 2.6". Disponível em: http://www.omg.org/cgi-bin/doc?formal/01-12-35. Acesso em 02/01/2004.

[10]Pereira F., "Televisão Digital: a Revolução já Começou!" Disponível em: http://si.porto.ucp.pt/internal/mestrado/mest99/Teoria_media/tv/TEXTOS%20FINAIS/Tv%20Digital.htm. Acesso em 02/02/2004.

[11]SAMBITS Project. System for Advanced Multimedia Broadcast and IT Services. Disponível em: http://www.darmstadt.gmd.de/delite/Projects/SAMBITS. Acesso em 02/01/2004.

[12]SUN Microsystems. Java Web Services. Disponível em: http://java.sun.com/webservices/webservicespack.html. Acesso em 02/02/2004

[13]SUN Microsystems. MIDP – Móbile Information Device Profile. Disponível em: http://jcp.org/aboutJava/communityprocess/final/jsr118/index.html. Acesso em 06/05/2004

[14]Synchronised and scalable AV content Across NeTwork. Disponível em: http://www.ist-savant.org/. Acesso em 21/08/2003.

[15]The Web Service Industry Portal. Disponível em: http://www.webservices.org. Acesso em 15/07/2003.

[16]Universal Mobile Telecommunications System. Disponível em : http://www.umts-forum.org/servlet/dycon/ztumts/umts/Live/en/umts/Home. Acesso em 21/08/2003.

[17]Waldo, J. "Javaspaces specification 1.0". Technical report (March), Sun Microsystems. March, 1998. 98. Feb 1996.

[18]Wap Forum. "Wireless Application Protocol Architecture Specification". Disponível em: http://www.wapforum.org/what/technical.htm

The Volterra representation of an electronic device using the Neural Network parameters

Georgina S. Stegmayer
Politecnico di Torino, Dipartimento Elettronica,
Torino, Italia, 10129
georgina.stegmayer@polito.it

and

Omar Chiotti
Universidad Tecnologica Nacional, GIDSATD,
Santa Fe, Argentina, 3000
chiotti@ceride.gov.ar

Abstract

Many electronic devices present nonlinear characteristics, which are often difficult to express analytically. Generally it is easier to perform measurements of the device parameters than to develop an analytical model of its behavior. As Neural Networks can be used to learn a system dynamics from input-output data only, we have developed a Neural Network model which reproduces the nonlinear behavior of an electronic device, in particular a Field-Effect Transistor (FET), using simulation data. However, electronic devices nonlinear analysis requires an analytical model (i.e. an equation representing the current-voltage relationship), described as a closed-form function, that allows to draw conclusions about the device, such as the Volterra series model. In this work, we want to show how the neural model and the analytical Volterra series model of the transistor are totally equivalent. Therefore, we show here how it is possible to build an analytical expression for a device nonlinearity, the Volterra series, with parameters of a standard Neural Network, trained with the device measurements or simulation data.
Keywords: Neural Networks, nonlinear electronic devices, Volterra model.

1. INTRODUCTION

Electronic device models based on nonlinear equivalent circuit for commercial CAD programs, generally require current-voltage and charge-voltage mathematical models, described as a closed-form function of the intrinsic control voltages, to characterize the nonlinear circuit elements. This equation describes the input/output behavior of the device/element. One example of such a function is a generally known and widely accepted transistor model: the Curtice Model [3].

One of the elements described by the Curtice Model is the nonlinear characteristic of the current in a Field-Effect Transistor (FET) device. The Curtice model equivalent circuit is presented in Figure 1. This model reflects the input/output behavior of the drain to source current *Ids*, that is function of the intrinsic voltages of the FET, *Vgs* and *Vds*. This element accounts for most of the nonlinear behavior of the device. We have performed some simulations to show the I/V (current vs. voltage) curves of the model (Figure 2), for different voltages combinations, using the following values: Vgs = [-1…0], Vds = [0…5], $\beta = 0$, $\chi = 0.3$.

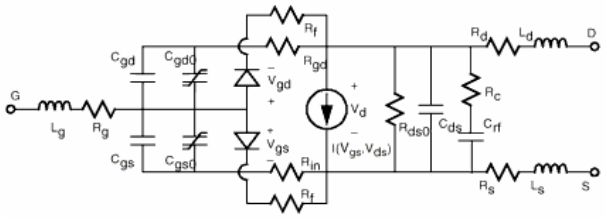


Figure 1. Curtice FET model equivalent circuit

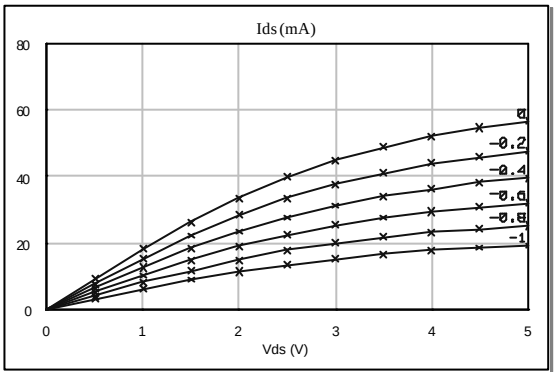


Figure 2. I/V Curves for the Curtice model

The model only shows the input/output characteristic of the element, but does not allow doing a deeper analysis about the device behavior that could help an electronic designer. One of the approaches regarding this analysis is the approximation of the nonlinear behavior of the element under consideration with a numerical series such as the Volterra series [4][2]. This series has some terms named *kernels* that allow a deeper understanding of the device. Its main disadvantage, however, is the analytical expression or calculation of its kernels. Our proposal is to use a neural network and its parameters, to help in the building of the Volterra series and the calculation of its kernels.

In Section 2 we present the Volterra series analysis and related work regarding the use of neural networks for Volterra kernels calculation. In Section 3 we present our neural network based model and how to calculate the kernels from parameters of the network. Simulations that confirm our proposal are shown in Section 4. The conclusions of the work can be found in Section 5.

2. VOLTERRA SERIES ANALYSIS

For a single-input, single-output (SISO) non-linear dynamical system, with an output time function, *y(t)*, and an input time function, *x(t)*, it can be represented exactly by a converging infinite series of the form

$$y(t) = \sum_{1}^{\infty} y_n(t) \tag{1}$$

$$y_n(t) = \int_{-\infty}^{\infty} \dots \int_{-\infty}^{\infty} h_n(t_1 .. t_n) \prod_{j=1}^{n} x(t - t_j) d t_j \tag{2}$$

This system can be represented to any desired degree of accuracy by a finite series of the form (3). This equation is known as the Volterra series expansion. The $h_1, h_2, \ldots, h_n$ are known as the Volterra *kernels* of the system [8]. The kernel h_0 is called the impulse response of the system, h_1 is the first order kernel, h_2 is the second order kernel, and in general, h_n is the n^{th} order kernel of the series. The Fourier transform of the kernel functions yields the transfer functions of the system. The transform of h_1 gives the linear transfer function and the transforms of the higher order kernels give the non-linear transfer functions.

$$y(t) = h_0 + \int_{t=0}^{\infty} h_1(t)x(t-t)dt + \int_{t_1=0}^{\infty}\int_{t_2=0}^{\infty} h_2(t_1,t_2)x(t-t_1)x(t-t_2)dt_1dt_2 + \ldots + \int_{t_1=0}^{\infty}\ldots\int_{t_n=0}^{\infty} h_n(t_1,t_2,\ldots,t_n)x(t-t_1)x(t-t_2)\ldots x(t-t_n)dt_1dt_2\ldots dt_n + \ldots \quad (3)$$

If the continuous Volterra series model (3) is discretized, then it assumes the form

$$y(k) = h_0 + \sum_{n=0}^{\infty} h_1(n)x(k-n) + \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty} h_2(n_1,n_2)x(k-n_1)x(k-n_2) + \ldots + \sum_{n_1=0}^{\infty}\ldots\sum_{n_n=0}^{\infty} h_n(n_1,n_2,\ldots,n_n)x(k-n_1)x(k-n_2)\ldots x(k-n_n) + \ldots \quad (4)$$

In nonlinear microwave analysis, in particular for small signal regime, the tool for excellence has been the Volterra-series analysis [8]. The Volterra description for an electronic device is based on Taylor-series expansions of the device nonlinearity around a fixed bias point, in the case of amplifiers or transistors, or around a time-varying waveform signal in the case of mixers. The nonlinear behavior in the device represented by the circuit of Figure 1, is particularly due to the drain to source current *Ids*. Its Volterra series expression truncated at the third order kernel is the following

$$\begin{aligned} Ids(Vgs,Vds) = {} & Ids(Vgs0,Vds0) + \\ & Gm1.vgs + Gds.vds + \\ & Gm2.vgs^2 + Gds2.vds^2 + Gmd.vgs.vds + \\ & Gm3.vgs^3 + Gds3.vds^3 + Gm2d.vgs^2.vds + Gmd2.vgs.vds^2 \end{aligned} \quad (5)$$

where *Vgs0* and *Vds0* are the internal bias voltages, *vgs = Vgs - Vgs0* and *vds = Vds - Vds0* are the incremental intrinsic voltages respect to the bias voltages. The coefficients of the series are the first order (*Gm1* and *Gds*), second order (*Gm2*, *Gds2* and *Gmd*) and third order (*Gm3*, *Gds3*, *Gm2d*, *Gmd2*) derivatives of the current. If we compare (4) and (5) we can see that these coefficients happen to be the Volterra kernels of the series.

These derivatives allow the inference of some device characteristics of great concern for the microwave designer, such as harmonic generation and inter-modulation phenomena in the case of a FET transistor, information that can be inferred from the derivatives of the model, which are contained in the Volterra kernels. For example, the coefficient *Gm1* is the first derivative of the current *Ids* with respect to the voltage *vgs*. It is an important parameter of the device called *transconductance*, which expresses the performance of the transistor

However, kernels calculation, measurement or analytical expression is a very complicated and time-consuming task [7] [1]. In the Biology field, these authors [13][12] have outlined a method for extracting the Volterra kernels of any order as a function of the weights and bias values of a feed-forward time delayed neural network with one hidden layer. Based on this idea, other works propose different strategies for kernels calculation with different neural networks topologies [11][9], also in the electronics field [5] [6]. But all of these approaches deal with time series inputs and only one variable.

However, in the case we are interested in, the function depends on two variables and the Volterra model is developed around two bias points, therefore the existing models cannot be used. We will propose a new model which can be used also in a multivariable case.

We have chosen to use a feed-forward multilayer perceptron (MLP) neural network, having the two intrinsic voltages as inputs, and the current as output. Differently from [4] where it is compared the use of two types of Radial Basis Function neural networks and an MLP to describe the current nonlinearity, where measurements of the current and all its derivatives are necessary for the model, which is a complex task; our approach, instead, only needs simple device measurements or simulations of the voltages and the output current, and the training of a very simple MLP neural network model. With only those elements, after performing some very simple calculation, the Volterra series and its kernels can be obtained.

3. NEURAL NETWORK BASED MODEL

The topology of the MLP network that we propose is a very simple one. It has one input layer, one hidden layer with two hidden neurons, and an output layer with a single output (Figure 3).

The input layer has two neurons, which are the controlled voltages *vgs* and *vds*. The inputs are multiplied by its corresponding weights w_{ij} (i,j=[1…2]) and propagated through the network. In the hidden layer there are two hidden neurons. As their activation function we have chosen the hyperbolic tangent (*tanh*). Each of them receives the sum of the weighted inputs, added to its corresponding bias value b_k (k=[1…2]). The output neuron has a linear activation function and therefore the output of the network *Ids* is calculated as the sum of the weighted outputs of the two hidden neurons plus a bias (6).

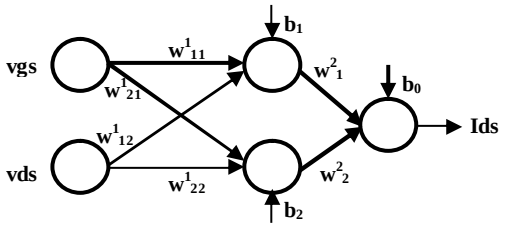


Figure 3. Topology of the MLP studied

$$Ids = b_0 + w_1^2\,[tanh(b_1 + w_{11}^1 vgs + w_{12}^1 vds)] + w_2^2\,[tanh(b_2 + w_{21}^1 vgs + w_{22}^1 vds)] \tag{6}$$

We want to show the equivalence between the output of our neural network model (6), that learns the relationship between the voltages and the current of the Curtice model, and its Volterra series representation (5), and we will show how easy is to obtain, from the network parameters, the values of the Volterra kernels that form the series.

Following the approach in [13], we expand the output of our network model (6) as a Taylor series around the bias values of the hidden nodes

$$Ids = b_0 + w_1^2 \sum_{j=0}^{\infty} \left[\frac{tanh^{(j)}(b_i)}{j!} \left(w_{11}^1 vgs + w_{12}^1 vds \right)^j \right] + w_2^2 \sum_{j=0}^{\infty} \left[\frac{tanh^{(j)}(b_i)}{j!} \left(w_{21}^1 vgs + w_{22}^1 vds \right)^j \right] \tag{7}$$

where $tanh^{(j)}$ is the *jth* derivate of the hyperbolic tangent (*tanh*).

Developing the brackets in (7) up to the second order derivatives and accommodating the common terms, yields

$$\begin{aligned}
&Ids = b_0 + \sum_{i=1}^{2} w_i^2\, tanh(b_i) + \\
&\left[w_1^2 w_{11}^1 (tanh^{(1)}(b_1)) + w_2^2 w_{21}^1 (tanh^{(1)}(b_2)) \right] vgs + \\
&\left[w_1^2 w_{12}^1 (tanh^{(1)}(b_1)) + w_2^2 w_{22}^1 (tanh^{(1)}(b_2)) \right] vds + \\
&\left[w_1^2 w_{11}^1 w_{11}^1 \left(\frac{tanh^{(2)}(b_1)}{2} \right) + w_2^2 w_{21}^1 w_{21}^1 \left(\frac{tanh^{(2)}(b_2)}{2} \right) \right] vgs^2 + \\
&\left[w_1^2 w_{12}^1 w_{12}^1 \left(\frac{tanh^{(2)}(b_1)}{2} \right) + w_2^2 w_{22}^1 w_{22}^1 \left(\frac{tanh^{(2)}(b_2)}{2} \right) \right] vds^2 + \\
&\left[w_1^2 w_{11}^1 w_{12}^1 \left(\frac{tanh^{(2)}(b_1)}{2} \right) + w_2^2 w_{21}^1 w_{22}^1 \left(\frac{tanh^{(2)}(b_2)}{2} \right) \right] vgs.vds + \ldots
\end{aligned} \tag{8}$$

Comparing (5) and (8) it can be seen that our developed neural network model is equivalent to the Volterra model or power-series expression commonly used for the current-voltage relationship in a FET. Moreover, the Volterra kernels of the series can now be easily calculated in function of the neural network parameters (the terms between brackets).

Equations (9) to (14) show how to calculate the kernels using the weights and bias values of the network. We only show the calculations for the impulse response, and the first and second order kemels, but the other formulas can be easily found. The corresponding first and second order derivatives have been developed.

$$Ids(Vgs0,Vds0) = h_0 = b_0 + \sum_{i=1}^{2} w_i^2 \tanh(b_i) \tag{9}$$

$$Gm1 = h_1(vgs) = \sum_{j=1}^{2} w_j^2 w_{j1}^1 (1 - \tanh^2(b_j)) \tag{10}$$

$$Gds = h_1(vds) = \sum_{j=1}^{2} w_j^2 w_{j2}^1 (1 - \tanh^2(b_j)) \tag{11}$$

$$Gm2 = h_2(vgs,vgs) = \frac{\sum_{j=1}^{2} w_j^2 w_{j1}^1 w_{j1}^1 (-2\tanh(b_j) + 2\tanh^3(b_j))}{2} \tag{12}$$

$$Gd2 = h_2(vds,vds) = \frac{\sum_{j=1}^{2} w_j^2 w_{j2}^1 w_{j2}^1 (-2\tanh(b_j) + 2\tanh^3(b_j))}{2} \tag{13}$$

$$Gmd = h_2(vgs,vds) = \frac{\sum_{j=1}^{2} w_j^2 w_{j1}^1 w_{j2}^1 (-2\tanh(b_j) + 2\tanh^3(b_j))}{2} \tag{14}$$

4. SIMULATION RESULTS

We have performed some simulations to show the validity of our approach. We have used the Curtice model to generate the training data, with simulations performed with an electronics circuits analyzer, but also laboratory measurements could have been used, because only the input/output data is necessary.

Once the neural network model was trained, using back-propagation and the Levenverg-Marquardt algorithms, to reproduce the nonlinear behavior of the system, we have extracted the weights and bias values from the neural network topology and have calculated the Volterra kernels of the system up to the third order, using the formulas shown above. Then we have built the Volterra approximation (8) with the calculated kernels, and we have plotted it against the original FET behavior.

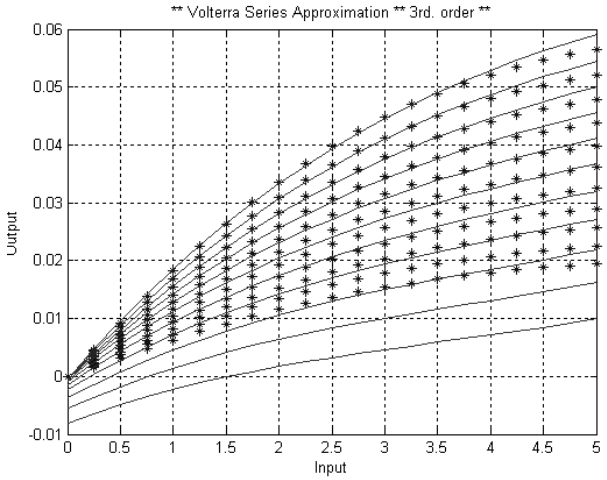


Figure 4. I/V Curves of the Curtice model (*) vs. I/V Curves of the Volterra-Neural Network based model (-), without normalized values

The results are reported in Figures 4 and 5. In Figure 4 we have trained the MLP model with the original input/output data, without normalizing the values. We can see that the Volterra approximation to the original data, built with the kernels calculated with the MLP model and including only up to the third order kernels, is quite accurate, but for the lower values of vgs there are some negative values which are not correct. However, in general, the form of the nonlinearity in this multivariable case is correct.

We have decided to try normalizing the input/output values to train the MLP model, to improve approximation, between the extremes of the domain interval of the hidden nodes activation function, the hyperbolic tangent. This way, the data are normalized before being used for the training, and de-normalized afterwards. The results have been impressively improved, and are shown in Figure 5, reaching a Mean Square Error (MSE) of $1x10^{-7}$.

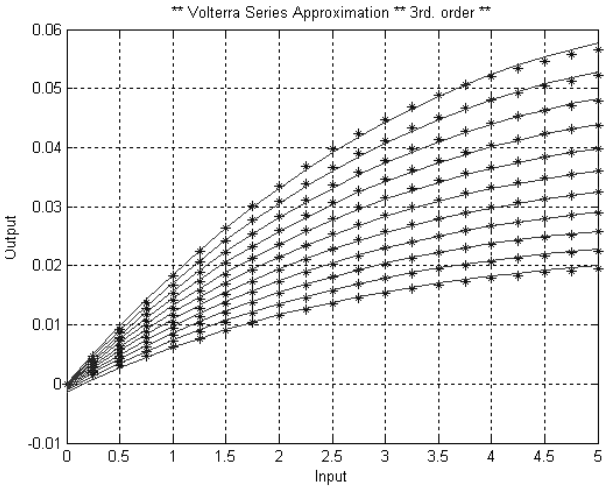


Figure 5. I/V Curves of the Curtice model (*) vs. I/V Curves of the Volterra-Neural Network based model (-) with normalized values

These results show the validity of our approach for the building of a multivariable Volterra series model of the nonlinearity in an electronic devices, and the results have shown also the importance of normalization for improving the Neural Networks training phase.

5. CONCLUSIONS

In this work, we have shown how a Neural Network model for a transistor nonlinearity and its analytical Volterra series model are totally equivalent. Therefore, we have explained here how it is possible to build the Volterra series analytical expression for an electronic device, even in the case of a multivariable nonlinearity, which is a difficult task, using parameters of a simple standard Neural Network model, trained only with voltage-current device measurements or simulation data.

From our simulation results we conclude that our approach is valid and that even when we have built a series having into account up to the third order kernels only, the approximation of the original function is quire accurate and fast. We have put in evidence also the importance of data normalization for the training phase on a neural network model, to improve accuracy in the final results obtained.

We want to highlight that our proposed approach implies and effective and concrete application of the neural networks models in the Electronics Field, that allows the building of an analytical Volterra series representation for a device, in this particular case a FET transistor, with the help of a very simple neural network model that needs few data, and some algebra, saving precious time to the microwave engineer at the moment of device analysis and design.

Our future work involves the implementation of this model inside a microwave circuit simulator.

References

[1] Bauer, A., Schwarz, W. Circuit analysis and optimization with automatically derived Volterra kernels. *IEEE International Symposium on Circuits and Systems*, (2000), I-491 - I-494.

[2] Boyd, S., Chua, L.O., Desder, C.A. Analytical Foundations of Volterra Series. *IMA Journal of Mathematical Control & Information*, No 1, (1984), 243-282.

[3] Curtice, W.R., Ettenberg, M. A Nonlinear GaAs FET Model for Use in the Design of Output Circuits for Power Amplifiers. *IEEE Transactions on Microwave Theory and Techniques*, Vol. 33, No. 12 (1985), 1383-1394.

[4] Garcia, J.A., Tazon Puente, A., Mediavilla Sanchez, A., Santamaria, I., Lazaro, M., Pantaleon, C.J., Pedro, J.C. Modeling MESFETs and HEMTs Intermodulation Distortion Behavior Using a Generalized Radial Basis Function Network. *International Journal on RF and Microwave Computer-Aided Design, Special number on Neural Networks*, No. 9, (1999), 261-276.

[5] Hakim, N.Z., Kaufman, J.J., Cerf, G., Meadows, H.E. Volterra characterization of Neural Networks. *Signals, Systems and Computers*, No. 2, (1991), 1128-1132.

[6] Harkouss, Y., Rousset, J., Chehade, H., Ngoya, E., Barataud, D., Teyssier, J.P. Modeling Microwave Devices and circuits for telecommunications systems design. *International Joint conference on Neural Networks*, No. 1, (1998), 128-133.

[7] Kashiwagi, H., Harada, H., Rong, L. Identification of Volterra kernels of nonlinear systems by separating overlapped kernel slices. *Conference of the Society of Instrument and Control Engineers of Japan*, No. 2, (2002), 707-712.

[8] Maas, S.A. *Nonlinear Microwave Circuits*. Ed. Artech House. 1988.

[9] Marmarelis, V.Z., Zhao, X. Volterra models and Three Layers Perceptron. *IEEE Transactions on Neural Networks*, Vol. 8, No. 6, (1997), 1421-1433.

[10] Rough, W. *Nonlinear System Theory. The Volterra/Wiener Approach*. Ed. Johns Hopkins University Press. 1981.

[11] Soloway, D.I., Bialasiewicz, J.T. Neural Networks modeling of nonlinear systems based on Volterra series extension of a linear model. *IEEE International Symposium on Intelligent Control*, (1992), 7-12.

[12] Woods, Q.P., Batchelor, A.H., Johnson, S.R., Green, G.G.R. Comparison of the response of a time delay NN with an analytical model. *International Joint Conference on Neural Networks*, No. 2, (2002), 1120-1125,

[13] Wray, J., Green, G.G.R. Calculation of the Volterra kernels of non-linear dynamic systems using an artificial neural network. *Biological Cybernetics*, No. 71, (1994), 187-195.

The 30th Latin-American Conference on Informatics

Carlos Araya Pacheco
Universidad Católica del Norte
Departamento de Ciencias Empresariales
Antofagasta,Chile.
caaraya@ucn.cl

Monique Olmos Carrasco
Universidad de Antofagasta
Departamento de Ingeniería de Sistemas
Antofagasta, Chile
molmos@uantof.cl

Predicción del Rendimiento de los Alumnos de Plan Común de las Carreras de Ingeniería Civil Industrial de la Universidad de Antofagasta a través de Minería de Datos.

Abstract:

The knowledge discovery in large databases is a process which can turn out competitive advantages to the companies, useful to their business models. Under this viewpoint, the outdoing of the information management is looked at as a must to the companies, in the never ending searching of newer and powerful useful behavioral patterns.

The goal of the research is to forecast the general achievement of the students who belong to junior industrial engineering program, in order to help the strategic objectives of the Faculty, giving it light about the successful criteria and factors related with, to set up the wished entry level behavior and help reducing the student drop-off and failing.

This research looks insight and describes the process, and was developed in the Engineering Faculty of the University of Antofagasta, Chile, using to evaluate the algorithms of the decision tree and the neuronal networks the methodology CRISP-DM.

The outcomes and conclusions of the research show a 95 % forecasting successful to the Calculus-I and Algebra-I subjects and a 70% forecasting successful to the school-grade scores and schooling type.

Árboles Decisión, Redes Neuronales, Minería Datos, Asociación de Reglas, Gestión Universitaria.

Introducción

En la actualidad, el descubrimiento de Conocimiento en Bases de Datos Masivas, es un proceso que puede entregar a las organizaciones educacionales ventajas competitivas para su modelo de negocio. Siendo fundamental el apoyo de los directivos en la definición de objetivos claros, los cuales guiarán esta búsqueda de conocimiento.

Existe en la comunidad científica muy pocos estudios del rendimiento de los estudiantes, con apoyo de técnicas de minería de datos, dentro de ellos se muestra una tendencia a definir el proceso como proyecto tecnológico o un proceso de gestión educacional.

En la primera visión, Dhammika [1] define subconjuntos de características que responden a rangos de variables asociados al tipo de organización. En la misma línea Salpeter [2]., valoriza la conducción de los datos, para cumplir los objetivos de Data Mining.

En cambio Thomas [3], plantea que la utilización de Data Mining es para identificar aspectos específicos de los estudiantes, que permita crea subconjuntos de sus características, utilizando árboles de decisión. Gornitza [4] plantea una reestructuración administrativa de las fuerzas de trabajo en las universidades, para impactar el rendimiento de los estudiantes a través de evaluaciones académicas.

En este contexto, esta investigación se basa en un proyecto desarrollado en la Facultad de Ingeniería, de la Universidad de Antofagasta (Chile), con la Metodología CRISP-DM, que presenta una metodología de procedimientos jerárquicos considerando una serie de tareas [5].

Las actividades a realizar en la fase de "comprensión del negocio", es entender los objetivos del proyecto y los requerimientos desde la perspectiva del negocio. En este caso se realizaron entrevista y reuniones con diferentes stakeholders de la Universidad de Antofagasta.

En segundo lugar se procedió a la "comprensión de los Datos", en la cual pretende obtener una familiaridad con los datos, descubriendo los primeros hechos o subconjuntos de datos interesantes, a fin de formular algunas hipótesis. Para ello se analizo el Modelo de Datos de la BD Simbad de Alumnos y Bienestar, perteneciente a la Universidad de Antofagasta.

Posteriormente en la tercera etapa de "Preparación de los Datos", se agregaron, eliminaron o modificaron algunos atributos de los datos para conformar el conjunto y subconjuntos de datos ha explorar. Se construyo un conjunto de datos de las notas de los alumnos y sus antecedentes más relevantes.

Una vez definido el conjunto de datos ha utilizar, se realizó el "Modelamiento", que consistió en la definición de las técnicas de modelamiento iniciales y finales, a través de múltiples secuencias de pruebas, hasta encontrar un modelo, con un grado de confianza y soporte adecuado. En este caso se utilizo las técnicas de Árboles de Decisión y Redes Neuronales

Finalmente conseguido un modelo adecuado de esta perspectiva para el análisis de los datos, se realizó la fase de la "Evaluación", donde se cerciora si también cumple con los objetivos de negocio, para su posterior implementación, en el caso puntual del proyecto, la evaluación esta siendo efectuada por los directivos de la Facultad de Ingeniería de la Universidad de Antofagasta.

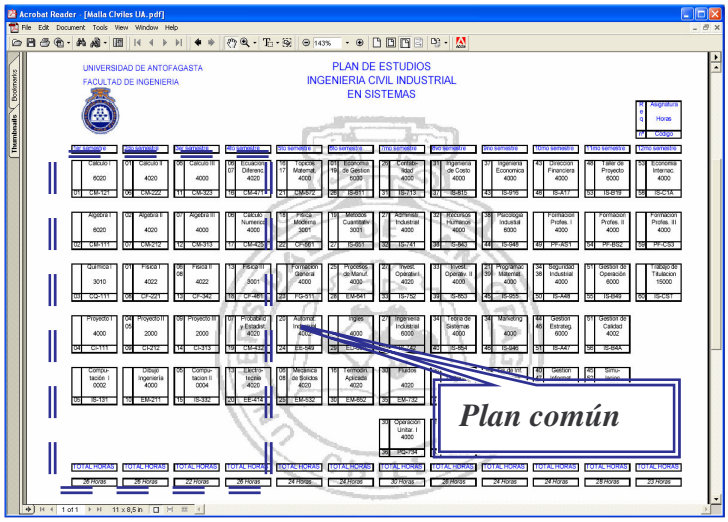
1. Comprensión del Negocio

En el inicio del Proyecto de Minería de Datos se debe individualizar la institución y los propósitos del negocio. En este caso particular se refiere a la Universidad de Antofagasta, específicamente a la Facultad de Ingeniería, en donde se imparten las Carreras de Ingeniería Civil Industrial, siendo el tronco común del cual se desprenden las Carreras de Ingeniería Civil Industrial Electricidad, Ingeniería Civil Industrial Electrónica, Ingeniería Civil Industrial Mecánica, Ingeniería Civil Industrial Minas, Ingeniería Civil Industrial Química y Ingeniería Civil Industrial Sistemas. Siendo el número de ingreso de alumnos a Ingeniería Civil Plan Común de aproximadamente 450 alumnos, los cuales después de 4 semestres académicos[1] en su avance curricular, deberán elegir las especialidades detalladas anteriormente.

El plan de estudios para los alumnos que optan a la Carrera de Ingeniería se puede visualizar a través de la siguiente malla curricular, donde se destaca el plan común con una línea punteada. (ver Figura 1).

[1] Semestre académico para Ingeniería: Se le denomina a un conjunto de 5 meses, los que pueden estar contenidos de marzo a julio para el primer semestre o de agosto a diciembre para el segundo semestre en la cual se dictan asignaturas, difiere para otras carreras donde las asignaturas son de marzo a diciembre.

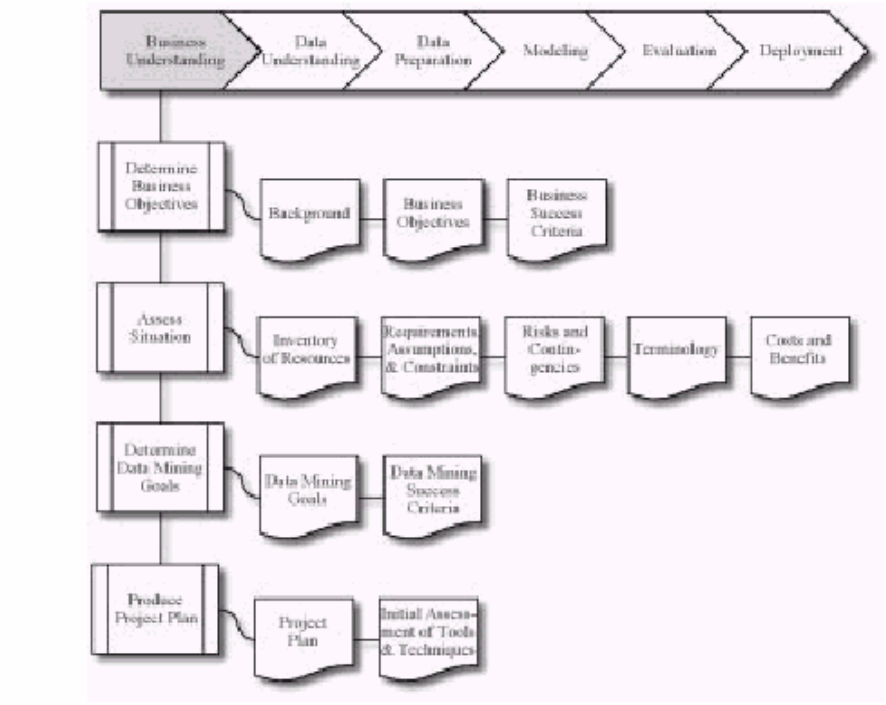
Figura 1. Plan de estudios de Ing. Civil Industrial, destacando área común.



Fuente: Web Site de la Universidad de Antofagasta

Para este proyecto se utilizó la Metodología Crisp-DM, que entrega una visión integral del proceso de Minería de Datos. En la Figura 2, se presenta un diagrama con los pasos a desarrollados durante el proyecto.

Figura 2: "Comprensión del Negocio". Modelo Crisp-DM 1.0



Fuente: Crisp-DM Step by Step Data Mining Guide. Página 16

1.1 Determinación de los Objetivos del Negocio

La Determinación de los objetivos de la investigación de Minería de Datos, esta relacionada con la comprensión del comportamiento de los alumnos en su ciclo básico, específicamente en cuales son los factores de éxito para superar el plan común de Ingeniería en la Universidad de Antofagasta, siendo una de las aristas de la evaluación. Además como complemento se necesitó analizar el rol del académico y los perfiles de egreso que se desean implementar a través del avance curricular, con un énfasis en un cluster de competencias definidas.

1.1.1 Background

La investigación se definirá como una "Predicción del Rendimiento de los alumnos en el Ciclo Básico de Ingeniería Civil Plan Común de la Universidad de Antofagasta", para definir los factores claves del éxito, que permitan definir las competencias de entrada al ciclo. Esta información podrá utilizarse para medir con mayor eficiencia los aspectos cuantitativos en el proceso de selección, tales como puntaje de ingreso y promedios de notas de enseñanza media, tipo de institución, etc.

1.1.2 Objetivos del Negocio

El Objetivo del negocio está definido en el Plan de Desarrollo de la Universidad de Antofagasta, que es una Institución de Educación Superior del Estado, independiente, autónoma y con personalidad jurídica propia, creada por Decreto Nº 11 del 10 de marzo de 1981. La Universidad de Antofagasta es la sucesora y continuadora legal de la Universidad de Chile y Universidad Técnica del Estado.

Dentro de la Declaración de la Misión Institucional se menciona en el punto 1,1 de los objetivos estratégicos, el "Aumento de la excelencia académica", como una línea de acción a desarrollar. Ello implica, por ejemplo, hacer el mayor esfuerzo por acreditar y ampliar la docencia de postgrado, y un esfuerzo importante para imbuir a los alumnos de pregrado con una clara conciencia de la calidad de la formación que reciben y mejorar su cultura general, y un esfuerzo razonable para mantener o incluso elevar la calidad de la docencia de pregrado.

Estos se resume en el punto 1.5 de la misión institucional , donde la definición de los factores claves de éxito para los alumnos de primer año de una carrera relevante dentro de la institución, tendrá un gran impacto en los indicadores definidos en los planes de acción. Considerando la inversión en aspectos que signifiquen una mejora cuantitativa y cualitativa del proceso de enseñanza, logrando la optimización de los recursos que siempre son escasos.

1.1.3 Criterios de Éxito en el Negocio

Los Criterios de Éxito, son en definitiva lo que determino el curso de la investigación a través de un análisis de las variables que afectan el proceso de enseñanza, considerando un periodo de 2 años en el Plan Común.

Para ello se realizó un estudio horizontal a través de líneas de conocimiento[2], que a continuación se detallan:

- Línea Cálculo: corresponde a Calculo 1, Calculo 2, Calculo 3, Ecuaciones Diferenciales, Calculo Numérico.
- Línea Álgebra: Álgebra 1, Álgebra 2, Álgebra 3, Probabilidad y Estadística, Ecuaciones Diferenciales.
- Línea Cispi: Proyecto1, Proyecto 2, Proyecto 3, Computación 1.
- Línea de Computación: Computación 1, Dibujo Ingeniería, Computación 2
- Línea Física: Química, Física 1, Física 2, Física 3.

[2] Líneas de Conocimiento: Se le denomina a un conjunto de ramos, que entre ellos tienen que cumplir un pre requisito para poder continuar con el siguiente.

Las definiciones de éxito, son dos, una de ellas es Éxito 1(Ex1), que son todos aquellos alumnos que tengas notas (nr1…nr10) iguales o mayores a 4, en las asignaturas de plan común. Ya que con estas condiciones cumplen con el requisito básico. Pero para obtener el Éxito 2 (Ex2), que es el conjunto óptimo, se utilizará la variable tiempo, en función al número de veces que los alumnos, deben cursar sus asignaturas obligatorias de segundo año (4 semestres), en un periodo menor o igual del tiempo máximo permitido (8 semestres). Este enfoque esta asociado a los objetivos de negocios de la economicidad de recursos, con la consiguiente disminución de costos en recursos humanos (académicos), debido a la alta repitencia, y los costos escondidos tales como ayudantes, instalaciones, becas, etc.

$$Ex1 = \{ nr_1, nr2, nr3, nr4, nr5, n6, nr7, nr8, nr9, nr10. / \frac{\sum nr_m}{10} \geq 4 \}$$

Donde:
Ex1= Éxito 1
nr1= Nota Álgebra1, nr2= NotaCalculo1, nr3= NotaQuimica1, nr4= NotaProyecto1, nr5= NotaComp1, nr6= NotaÁlgebra2, nr7= NotaCalculo 2, nr8=NotaFisica1, nr9=NotaProyecto2, nr10= Computación2.

El éxito 2 se descompone en 5 éxitos individuales por las líneas de conocimiento, asociadas a la sumatoria de la cantidad de veces que el alumno realiza la asignatura, para llegar con éxito al último ramo de la línea.

$Ex2 = \{ Ex2ecuadif , Ex2calnum , Ex2fis3, Ex2prob, Ex2proy, Ex2comp2 / \Sigma Ex2 \leq 48 \}$

$Ex2ecuadif = \{Nalg1 , Nalg2, Nalg3, Nedif, Ncal1, Ncal2, Ncal3 / \Sigma Nm \leq 8\}$

$Ex2calnum = \{Nalg1 , Nalg2, Nalg3, Ncnum, Ncal1, Ncal2, Ncal3 / \Sigma Nm \leq 8\}$

$Ex2fis3 = \{Ncal1 , Nqui, Nfis1, Nfis2, Nfis3, Ncnum / \Sigma Nm \leq 8 \}$

$Ex2prob = \{Nalg1 , Nalg2, Nalg3, Nprob / \Sigma Nm \leq 8\}$

$Ex2proy = \{Nproy1 , Nproy2, Nproy3, Ncomp1 / \Sigma Nm \leq 8\}$

$Ex2comp2 = \{ Ncomp1, Ndibing, Ncomp1 / \Sigma Nm \leq 8\}$

Donde
Nalg1 =Numero Veces Algebra1, Nalg2 =Numero Veces Álgebra 2, Nalg3 =Numero Veces Álgebra 3, Nedif = Numero Veces Diferenciales, Nprob = Numero Veces Probabilidad, Ncal1 = Numero Veces Calculo1, Ncal2 = Numero Veces Calculo2, Ncnum = Numero Veces CalculoNumerico , Nproy1=Numero Veces Proyecto1 , Nproy2= Numero Veces Proyecto2, Nproy3= Numero Veces Proyecto3, Ncomp1= Numero Veces Computación1 , Ncomp2=Numero Veces Computación2, Ndibing= Numero Veces DibujoIngeniería, Nfis1= Numero Veces Fisica1, Nqui= Numero Veces Química, Nfis2= Numero Veces Fisica2, Nfis3= Numero Veces Fisica3.

1.1.4 Riesgos y Contingencias

Los riesgos intrínsecos que estaban asociados, fueron un elemento a evaluar en los procesos de gestión del proyecto. En el caso particular el trabajo con Bases de Datos nos presento los siguientes riesgos inherentes

1. Inconsistencia de Datos (formato): Los Datos originales del modelo de datos, no coincidía con las utilizadas actualmente para la gestión de la información. Se realizaron procesos de evaluación y comprensión de datos
2. Incompatibilidad de Bases de Datos: Las bases de Datos que se manejaban en el Departamento de Informática y Admisión, no eran las mismas, por esta razón se utilizaron los datos de Informática, que contenían mayor cantidad de registros históricos.

1.2 Determinación de Metas de Minería de Datos

La investigación considero como metas de negocio ha satisfacer, la disminución de la cantidad de deserciones y repitencia de las asignaturas del Plan Común, que esta acordes a una institución sin fines de lucro, como es la Universidad de Antofagasta, buscando definir patrones de conducta e identificando cuales son las variables que más influencia en las distintas asignaturas.

1.2.1 Metas de Minería de Datos

En cuanto a metas específicas del proyecto se definió como medida cuantitativa los siguientes aspectos

1. Definir el rango de puntajes mínimos como conducta de entrada de los alumnos a la carrera de Ingeniería Plan Común
2. Definir las variables de mayor influencia en el comportamiento de los alumnos en las asignaturas de primer año de la carrera de Ingeniería Plan Común (Factores Críticos de Éxito)

2 Comprensión de los Datos

2.1 Colección Inicial de Datos

2.1.1 Reporte de la Colección de Datos

Los datos utilizados para el presente proyecto, como colección inicial fueron entregados por la Dirección y Administración de Registro Curricular (DARC)[3] de la Universidad Antofagasta, el modelo de datos a utilizar fue entregado por el Departamento de Informática[4].

Dentro de los Criterios de selección de los datos, podemos enumerar los siguientes :

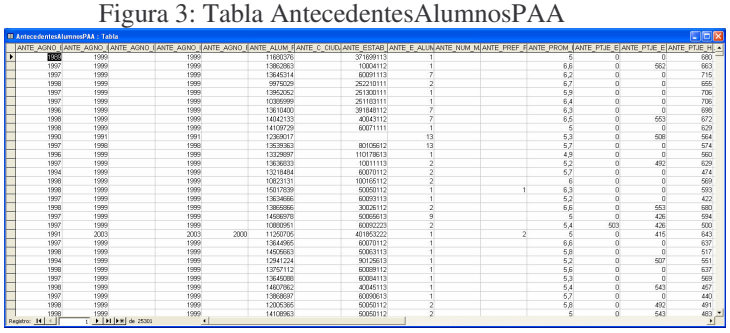
- Los datos de los alumnos que ingresa vía PAA[5] a la carrera de Ingeniería
- Datos históricos de 5 años de Ingresos de alumnos.
- Seguimiento curricular correspondiente a dos semestres, con un máximo de tiempo de aprobación (doble de la cantidad de años).

2.2 Descripción de los Datos

2.2.1 Reporte de Descripción de los Datos

En la mayoría de las tablas, la información estaba incompleta o simplemente era inexistente. El volumen de datos ascendían a 25.301 (ver figura 3).

Figura 3: Tabla AntecedentesAlumnosPAA



Fuente: Adaptación de Tabla en Access

[3] DARC: Departamento de Administración y Registro Curricular

[4] Departamento de Informática: Dentro de la Universidad Antofagasta esta inserto en el área de planificación, uno de los objetivos es mantener funcionando en forma optima los diferentes Sistemas de Información con que cuenta la Universidad, como así también mantener los Sistemas en líneas.

[5] Prueba Actitud Académica (PAA): Es un examen que evalúa las aptitudes de los alumnos que han terminado la Enseñanza Media, para la Facultad de Ingeniería esta evaluación debe ser superior a 450 puntos para poder postular (cuerdo tomado por Consejo de Rectores de Chile para las Universidades que pertenecen a el)

En una primera etapa se consideraron para el análisis preliminar, las tablas de Anteced_Postul (considera los datos de ingreso de los alumnos, p.e. puntaje de PAA), la tabla NotasRamos(considera las notas de los alumnos de todas las carreras de los años 1999 al 2001). Para este efectos se realizaron diversas Querys, para obtener la información de los alumnos de Ingeniería Civil, referente a los cuatros semestres con sus respectivas asignaturas y los datos de la PAA (análisis horizontal) y las notas de las líneas de conocimiento con los datos de la PAA (análisis vertical). Ambos análisis se consolidaron en la ConsultaFinal.

Los datos recogidos en la consulta final de Access, fueron consolidados, para finalmente migrar la tabla a un archivo en el Software SPSS versión 11.0, llamado ConsultaFinal.SAV .

2.3 Exploración de los Datos

Para iniciar la exploración de los datos, se importo el archivo ConsultaFinal.SAV al software de Data Mining SPSS Clementine, versión 6.5, herramienta que permite realizar los estudios estadísticos de los datos de acuerdo a su tipo y para cada campo, obteniendo ocurrencias, media, desviación estándar, error típico de la media y correlaciones.

En esta etapa de exploración primero se verifico la calidad de los datos, del total de registro del universo en estudio (2887 registros), obteniendo un 100 %.

En una segunda etapa se realizo los estudios estadísticos, para todos los campos, entregando información relevante sobre las desviaciones estándar y correlaciones, que en muchos casos permitieron encontrar relaciones interesantes entre los datos, que posteriormente se transformaron en hipótesis a confirmar.

Una de las hipótesis planteadas, se relaciona con las líneas de Cálculo y Álgebra, debido a la presencia de una correlación positiva alta, la cual es presente en todas las asignaturas de ambas líneas, y que se relacionan con las deserciones y la eliminación.

En paralelo en el Software SPSS 11.0 se amplio el estudio, para obtener las frecuencias que se presentan en los datos. El análisis se efectuó para cada unos de los campos, y se complemento con histogramas

3 Preparación de los Datos

3.1 Selección de los datos

Finalmente en la exploración de los datos se definió iniciar la selección para el proceso de minería de datos, considerando las metas de éxito 1 y éxito 2, para la hipótesis de la implicancia del éxito de las líneas de cálculo y álgebra. Para este efecto se definieron 2 conjuntos de 3 muestras para probar el modelo de 1000 registros cada una. Para el primer conjunto se definió el criterio de 1 de cada 2 registros, 1 de cada 3 registros, 1 de cada 4 registros y para el segundo conjunto se crearon 3 grupos de registros con una selección aleatoria.

3.2 Limpieza de Datos

La limpieza de datos es un proceso, en el cual se utilizan una gran cantidad de recursos, según Piramuthu [6], corresponden a un 80%, en el caso del proyecto tuvo una incidencia del 60% del tiempo con un costo asociado del 50 %. Los datos seleccionados, tenían un proceso de selección (querys), para tener solo los alumnos del plan común, como metodología de limpieza se eliminaron los alumnos que no tenían información de las asignaturas, y aquellos que no tenían cursadas las asignaturas se rellenaron los campos con valores ceros, para diferenciarlos de los alumnos que cursando la asignatura reprobaron con notas 1. La misma regla se utilizo para los campos relacionados con la Prueba de Aptitud Académica.

3.3 Construcción de Datos

En esta etapa se derivaron nuevos atributos, para preparar los datos para los algoritmos seleccionados, que para este caso, son los árboles de decisión y redes neuronales en función al problema de negocio presentado, que se puede definir como predictivo.

Este proceso de definición se realizo para que el algoritmo de árbol de decisión pueda generar un modelo más óptimo, a través de una derivación en un campo de tipo booleano. Esto entrega un grupo de alumnos que cumplen con parte de lo definido en éxito 1, que sirve para determinar la constante disminución en la cantidad de alumnos que pueden llegar al final del plan común.

En la exploración y manipulación de Datos surgieron algunas hipótesis, que se esperaban probar a través de las técnicas de modelado a utilizar, dichas técnicas necesitaron que se derivaran campos, es por esto que los campos iniciales se filtraron para no utilizarlos dos veces en el algoritmo.

4 Modelado

4.1 Selección de la técnica elegida

Las técnicas elegidas en función al problema de negocio presentado, inicialmente son:

a. Las redes neuronales
b. Los árboles de decisión

Estas técnicas se pueden definir como predictivos, con datos que son discretos (notas) y datos continuos (PAA), por ende un algoritmo de clasificación puede examinar las características de un nuevo alumno y asignarlo a una clase dentro de comportamiento de sus notas para el Plan Común.

4.2 Generación de diseño de prueba

El entrenamiento de los datos de prueba se presentará a través de las dos técnicas previstas, con las muestras seleccionadas en el modelo:

a. red neuronal
b. árboles de decisión

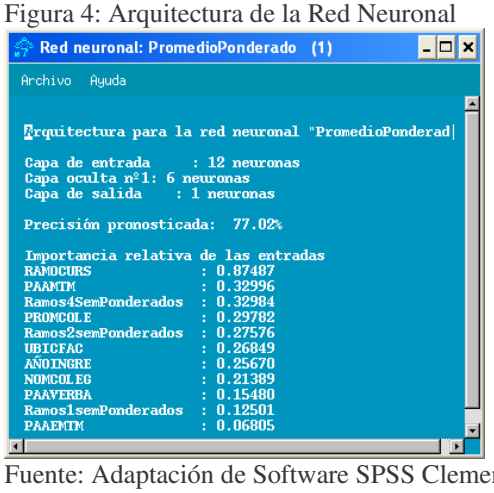
4.2.1 Red neuronal.

El entrenamiento de prueba de las muestras se ha realizado tras ejecutar el algoritmo de la red neuronal C5. Con ello obtendremos un modelo, que se espera sea potencialmente útil para el usuario final.

La información sobre la topología de la red, las estimaciones y la exactitud se puede visualizar en la figura 19, ella nos indica el número de neuronas en las capas de entrada, teniendo solo una salida. Así mismo la precisión es de un 77,02 %, para la salida PromedioPonderado.

En el análisis de sensibilidad, se tiene que el campo RamoCurs tiene una mayor importancia relativa con un 87%, lo sigue el puntaje de la PAA Matemáticas con una importancia relativa de un 32%.

Figura 4: Arquitectura de la Red Neuronal



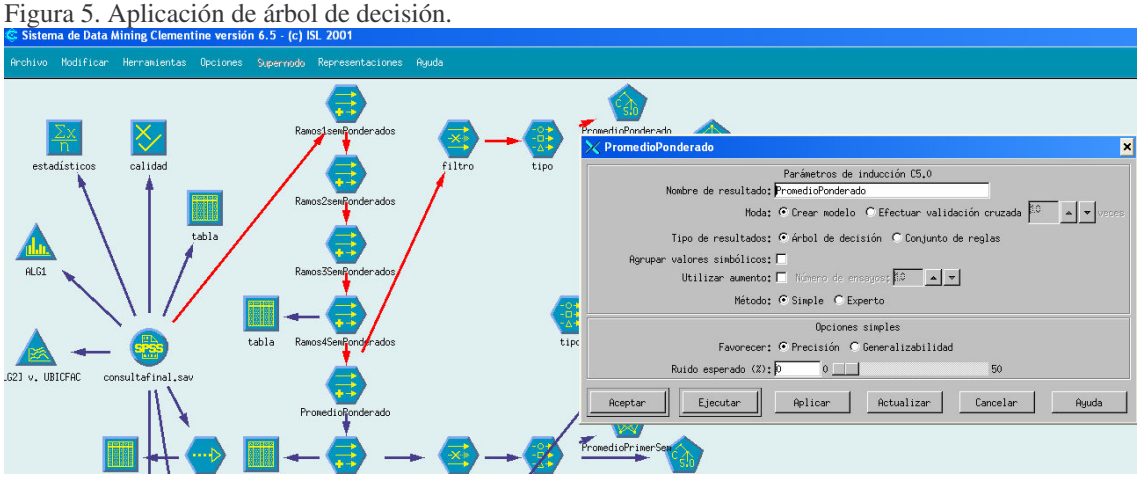
Fuente: Adaptación de Software SPSS Clementine

4.2.2 Árboles de Decisión.

El entrenamiento de los datos de prueba a través de la técnica árbol de decisión C5, en un proceso de descubrimiento supervisado, esta basado en los campos que nos proporcionarán el máximo de información posible (para el PromedioPonderado), esta nos entregará un conjunto de reglas, que para el usuario final sean potencialmente útiles. En esta etapa se evaluaron diversos algoritmos, optando por el C5, siendo el que ofrecía un mejor rendimiento, según Osei [7], esta fase de analizar diversos algoritmos, se podría automatizar a través de un proceso de multi-criterios.

En la figura 5, se muestra la ejecución (a través de la secuencia de flechas rojas) del algoritmo de Árbol de Decisión C5.

Figura 5. Aplicación de árbol de decisión.

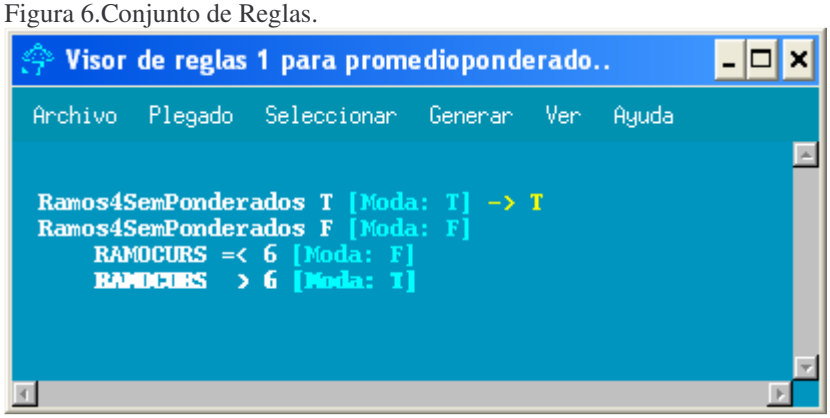


Fuente: Adaptación de Software SPSS Clementine

Tras aplicar el algoritmo, para construir un árbol de decisión C5, este nos da un conjunto de reglas (ver Figura 6), se puede interpretar que cuando el alumno cursa a los menos 6 ramos es muy factible que logre aprobar la totalidad de los ramos de plan común.

Los parámetros de entrada son los que se han definido en el punto 3.4 y se visualizan en la figura 6.

Figura 6.Conjunto de Reglas.



Fuente: Adaptación de Software SPSS Clementine

Se puede mencionar que respecto a los rendimientos a las variable de tiempo, los árboles de decisión son mucho más eficientes y entregan un resultado más claro para el usuario final.

5 Evaluación

Finalmente los algoritmos presentaron el siguiente conjunto de regla (se presentan las más relevantes) para Éxito 1,Éxito 2.

Regla 1 : Si en PAA Matemática >= 657 el Promedio Final Ponderado será verdadero.
Regla 2: Si en Promedio Colegio >= 6,1 el Promedio Final Ponderado será verdadero
Regla 3: Si Nota Colegio >= 5,6 Y PAA Matemática >=614 el Promedio Final Ponderado será verdadero
Regla 4: Si Álgebra 1 >=4 entonces Calculo 1 será Aprobado.

Donde: Éxito 1 = Promedio Final Ponderado.

Para las reglas anteriormente mencionadas, el análisis de los resultados son los siguientes:
Regla 1 → 61% Datos Correctamente Clasificados
Regla 2 → 70% Datos Correctamente Clasificados
Regla 3 → 65% Datos Correctamente Clasificados
Regla 4 → 95% Datos Correctamente Clasificados.

6 Consideraciones Finales

La investigación de Predicción del Rendimiento de los Alumnos de la Facultad de Ingeniería de la Universidad de Antofagasta, a través de Minería de Datos, nos permite presentar las siguientes consideraciones finales:

i. Existe una relación importante entre las asignaturas de Cálculo y Álgebra, que inciden directamente en el rendimiento de los estudiantes de ingeniería. Debido a que la asignatura de Cálculo es el prerrequisito más importante para acceder a las asignaturas finales de cuarto semestre de Ecuaciones Diferenciales, Calculo Numérico, Física III, que corresponde a un 60% de los requisitos para éxito 1, con capacidad de predicción del 95% de los casos.

ii. Las notas y el tipo de colegio, tiene un alto grado de incidencia en el rendimiento de los estudiantes, con una capacidad de predicción del 70 % de los casos.

iii. Los resultados relacionados a la Prueba de Aptitud Académica, no necesariamente pueden presentar resultados similares con la actual PSU.

7 Referencias

[1] Dhammika, A. Mining data to find subset of high activity. Journal of Statistical of Planing & Inference.Vol. 122 Issue ½ , (May 2004), pp 19-34.

[2] Salpeter, J. Data: Mining with a Mission. Technology & Learning. Vol. 24, Issue 8, (March 2004), pp30-36.

[3] Thomas, E. What Satisfies Students? Mining Student-Opinion Data with Regression and Decision Tree Analysis. Vol. 45, Issue 3, (May 2004), pp 30-36.

[4] Gornitzka, A. Towards professionalisation ? Restructuring of administrative work force in universites. Vol. 47, Issue 4, pp 445-472.

[5] Chapman, P. Crisp DM 1.0 Step by Step Data mining guide. Crisp DM Consortium.

[6] Piramuthu, S. Evaluating feature selection methods for learning in data mining aplications. European Journal of Operational Research. Vol. 156 Issue 2, pp 483-495.

[7] Osei B. Evaluation of decision trees: a multi-criteria approach. Vol 31. Issue 11, (September 2003), pp 1933-1946.

Estudo da Viabilidade de Utilização o *Framework* GREN para Instanciar Aplicações no Domínio de Clínicas de Reabilitação

Anderson Pazin[1]
Faculdades Salesiana de Lins, Centro de Tecnologia de Informação.
Lins - SP, Brasil, CEP: 16400-505, 55(0xx14) 3522 - 4733
anderson@salesianolins.br

Ricardo Argenton Ramos
Rosângela Ap. D. Penteado
[1]UFSCar - Universidade Federal de São Carlos, Departamento de Computação
São Carlos - SP, Brasil, CEP:13565-905, 55(0xx 16) 260-8233
{ rar, rosangel}@dc.ufscar.br

Abstract

The domain of rehabilitation clinics management can be considered as a sub domain of bussiness resource management. The GREN framework instanced applications of the domain bussiness resource management using the pattern language GRN. However, such framework does not deal with specific functionalities of the domain of rehabilitation clinics management, as for example the accompaniment of the treatment at patient. A pattern language for this domain, called SiGCli (Sistemas para Gerenciamento de Clínicas in protuguese) was elaborated on the basis of the GRN. Some patterns of the GRN had been adapted to the specific necessities of the SiGCli domain. This paper presents the use of GREN framework to instanced applications of the domain of rehabilitation clinics management, being commented the advantages and disadvantages of this use.

Key Words: pattern language, management of rehabilitation clinics, bussiness resource, reuse.

Resumo

*O domínio de gerenciamento de clinicas de reabilitação, pode ser considerado como um subdomínio de gestão de recursos de negócios. O framework GREN instancia aplicações do domínio de gestão de recursos de negócios utilizando a linguagem de padrões GRN. Entretanto, tal framework não trata de funcionalidades específicas do domínio de gerenciamento de clinicas de reabilitação, como por exemplo o acompanhamento do tratamento de um paciente. Uma linguagem de padrões para esse domínio, denominada SiGCli (**Si**stemas para **G**erenciamento de **Cli**nicas de reabilitação) foi elaborada com base na GRN. Alguns padrões da GRN foram adaptados às necessidades específicas do domínio SiGCli. Este trabalho apresenta a utilização do framework GREN para instanciar aplicações do domínio de clinicas de reabilitação, comentando as vantagens e desvantagens dessa utilização.*

Palavras-Chave: linguagem de padrões, gerenciamento de clínicas de reabilitação, gestão de negócios, reuso.

.

1. Introdução

O reuso de software é uma atividade comum em empresas de desenvolvimento, devido à necessidade de melhorar a produtividade, a manutenibilidade e a qualidade tanto do software quanto do processo de desenvolvimento. Uma forma de melhorar o reuso é utilizar padrões de software, por possibilitarem o reuso em vários níveis de abstração (código, projeto, análise, arquitetura e processo de desenvolvimento) [2].

Padrões de software descrevem soluções de sucesso para os problemas de desenvolvimento de software que geralmente ocorrem em determinados contextos, com a finalidade de auxiliar os desenvolvedores menos experientes com os conhecimentos dos mais experientes, para que possam agir como especialistas [3].

Uma linguagem de padrões é uma coleção de padrões organizados que se apóiam entre si para transformar requisitos e restrições numa arquitetura [4]. Os padrões que a constituem devem abranger todos os aspectos importantes de um determinado domínio e pelo menos um padrão deve estar disponível para cada aspecto da construção e implementação de um sistema de software. A linguagem de padrões auxilia na subdivisão de problemas gerais com soluções complexas em problemas menores e relacionados, de forma a facilitar a solução. Uma característica importante para as linguagens de padrões é que cada padrão pode ser usado de forma isolada ou com um certo número de padrões da linguagem, com isso um único padrão é considerado útil mesmo que a linguagem não seja aplicada em sua totalidade.

Um *framework* é uma infra-estrutura genérica, baseada em um domínio, que pode ser adaptada para solucionar problemas específicos desse domínio, servindo como um modelo para a construção de aplicações através da especificação das classes e das colaborações entre elas. Um *framework* reusa: a) análise, pois descreve os tipos de objetos importantes e como um problema maior pode ser dividido em problemas menores; b) projeto, pois contém algoritmos abstratos e descreve a interface que deve ser implementada, e as restrições a serem satisfeitas pela implementação; e c) código, pois se torna mais fácil desenvolver uma biblioteca de componentes compatíveis e porque a implementação desses pode herdar grande parte de seu código das super-classes abstratas. Apesar de todos esses tipos de reuso serem importantes, os reusos de análise e de projeto são os que mais apresentam vantagens em longo prazo [5,6].

Esse artigo apresenta os resultados obtidos ao se aplicar uma linguagem de padrões específica (SiGCli), definida para um subdomínio de gestão de recursos de negócios, utilizando o framework GREN,desenvolvido com base na linguagem de padrões GRN, para instanciar aplicações desse subdomínio. A seção 2 apresenta a linguagem de padrões GRN e o *framework* GREN, a seção 3 trata do processo de elaboração a linguagem de padrões SiGCli [7] com base em outra linguagem de padrões. A aplicação dos padrões da SiGCli com *framework* GREN é mostrada na seção 4 e as considerações finais sobre o estudo realizado são apresentadas na seção 5.

2. Assuntos Relacionados

2.1. A Linguagem de Padrões GRN.

A Linguagem de Padrões para Gestão de Recursos de Negócios (GRN), [8,9,10] é composta por quinze padrões de análise, sendo alguns deles aplicações ou extensões de padrões existentes na literatura. Essa linguagem auxilia desenvolvedores menos experientes no desenvolvimento de aplicações que tratam de gestão de recursos de negócios, ou seja, nas quais existe a necessidade de registrar transações de aluguel, comércio ou manutenção de recursos. A Figura 1, extraída de [10] exemplifica a GRN, mostrando os relacionamentos e dependências entre os padrões. A ordem de aplicação desses padrões é definida, bem como as dependências existentes entre eles, representadas pela descrição do elemento "Próximos Padrões". Na Figura 1, as linhas mais espessas indicam os principais padrões da linguagem: LOCAR O RECURSO, COMERCIALIZAR O RECURSO e MANTER O RECURSO.

Os padrões estão agrupados de acordo com seu propósito. No primeiro grupo estão três padrões: IDENTIFICAR O RECURSO, QUANTIFICAR O RECURSO e ARMAZENAR O RECURSO, que tratam da identificação e possível qualificação, quantificação e armazenagem dos recursos gerenciados pelo negócio. No segundo grupo estão os padrões relacionados à manipulação dos recursos de negócio pelo sistema. Existem sete padrões nesse grupo: LOCAR O RECURSO, RESERVAR O RECURSO, COMERCIALIZAR O RECURSO, COTAR O RECURSO, CONFERIR A ENTREGA DO RECURSO, MANTER O RECURSO e COTAR A MANUTENÇÃO. O terceiro grupo possui cinco padrões que cuidam de detalhes das transações efetuadas com o recurso. Os três primeiros, ITEMIZAR TRANSAÇÃO DO RECURSO, PAGAR PELA TRANSAÇÃO DO RECURSO e IDENTIFICAR O EXECUTOR DA TRANSAÇÃO, são aplicáveis a quaisquer das transações do grupo 2. Os dois últimos, IDENTIFICAR AS TAREFAS DA MANUTENÇÃO e IDENTIFICAR AS PEÇAS DA MANUTENÇÃO, são aplicáveis às transações contidas nos padrões MANTER O RECURSO e COTAR A MANUTENÇÃO.

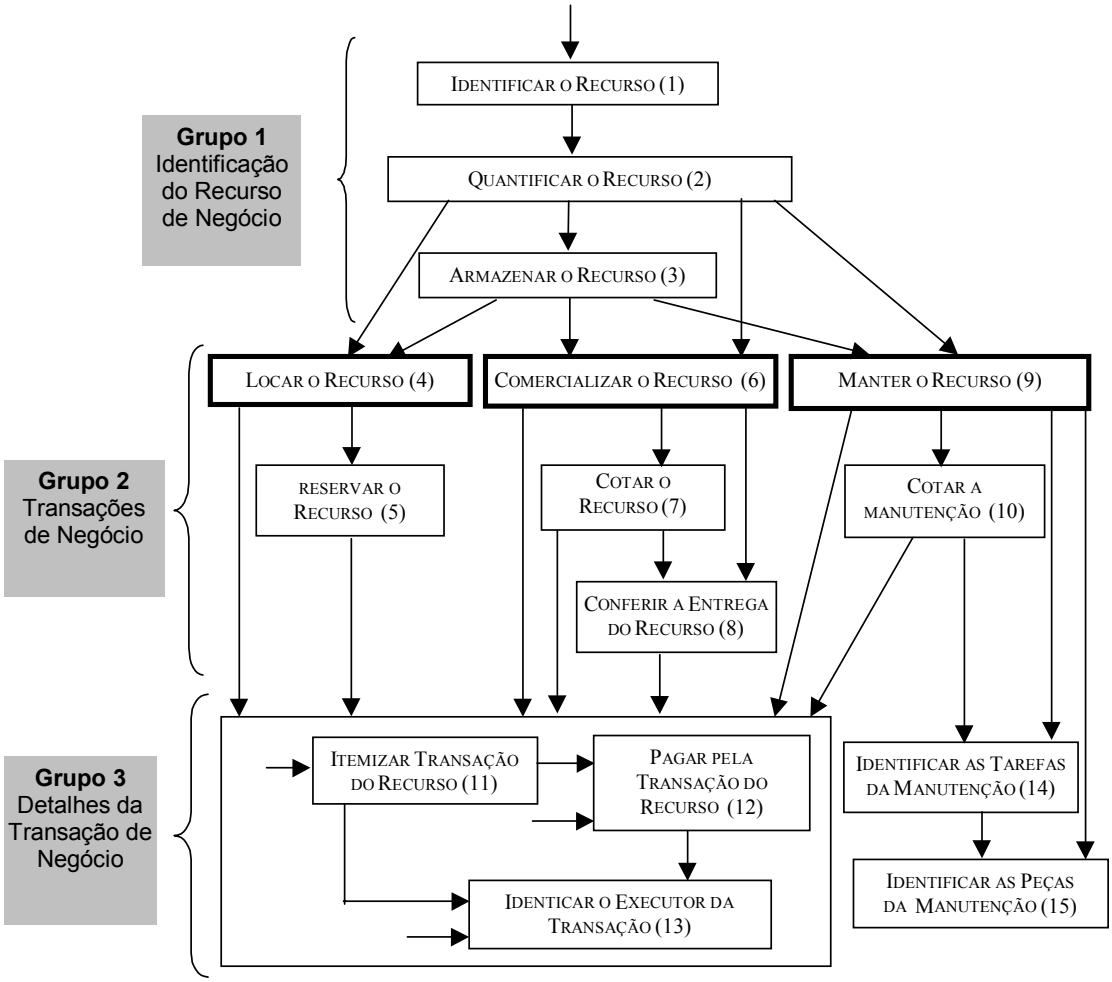


Figura 1. Relacionamento entre os padrões da linguagem GRN [10]

2.2. O *Framework* GREN.

Braga e Masiero [1] propõem um processo de construção de *frameworks* com base em uma linguagem de padrões. Dessa forma, a GRN foi utilizada para elaborar o *framework* GREN, desenvolvido em linguagem *Smalltalk*, para instanciação de aplicações no domínio de sistemas de gestão de recursos de negócios. A Figura 2 mostra a arquitetura do *Framework* GREN.

A camada *GREN-Persistência* possui classes para cuidar da conexão com a base de dados, gerenciamento dos identificadores dos objetos e persistência dos objetos. A camada *GREN-Negócios* comunica-se com a camada *GREN-Persistência* sempre que for necessário armazenar permanentemente um objeto. Dentro da camada *GREN-Negócios* existem diversas classes derivadas diretamente dos padrões de análise que compõem a GRN, ou seja, as classes e os relacionamentos contidos em cada padrão possuem a implementação correspondente nessa camada. A camada *GREN-Interface Gráfica com o Usuário* contém as classes responsáveis pela entrada e saída de dados, como formulários de interface, janelas e menus, que permitem a interação do usuário final com o sistema. Essa camada comunica-se com a *GREN-Negócios* para obtenção de objetos a serem mostrados na interface com o usuário. A camada *GREN-Wizard*, encontra-se acima da camada de interface gráfica com o usuário, pois utiliza todas as demais camadas para realizar a instanciação de aplicações de forma gráfica com base na GRN.

As aplicações específicas podem ser instanciadas a partir da camada *GREN-Interface Gráfica com o Usuário* usando, por meio de herança ou referência a objetos, classes de todas as camadas do GREN. Quando não existem classes a serem reutilizadas da camada *GREN-Interface Gráfica com o Usuário*, a instanciação de uma aplicação é realizada a partir da *GREN-Negócios*, sendo a interface implementada separadamente. Outra opção é a partir do *GREN-Wizard*, que se encarrega de fazer a comunicação com as demais camadas do GREN.

As instanciações das aplicações no GREN são guiadas pela aplicação da Linguagem de Padrões GRN, que gera diagramas de classes da aplicação desejada. Com base nesses diagramas utilizam-se classes pré-programadas do *framework* GREN para criar o código da nova aplicação.

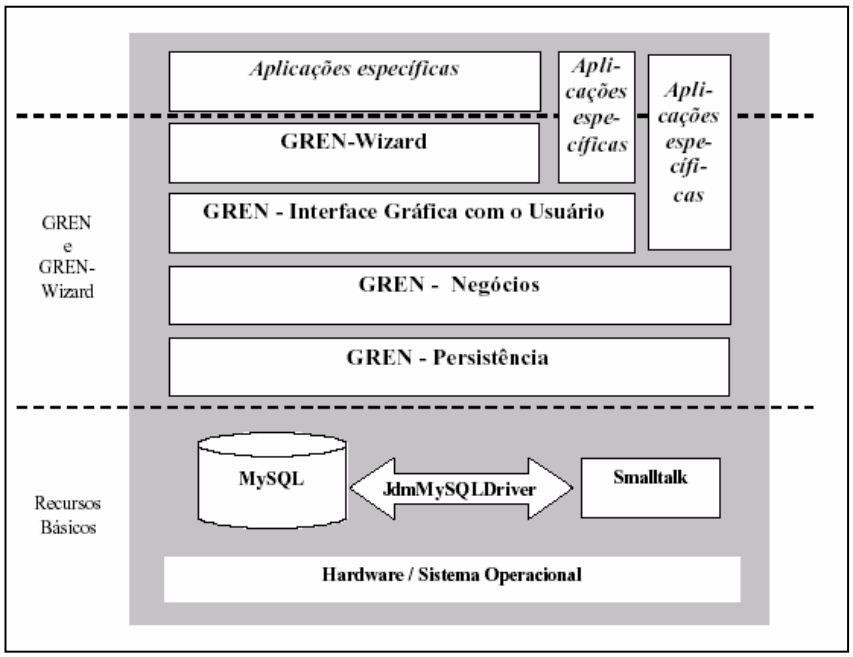


Figura 2. Arquitetura do Framework GREN [10]

3. A Linguagem de Padrões SiGCli

A linguagem de padrões para **Si**stemas de **G**erenciamento de **Cl**ínicas de Reabilitação, denominada SiGCli, foi elaborada com base no mesmo processo usado para a elaboração da GRN [10], Figura 3.

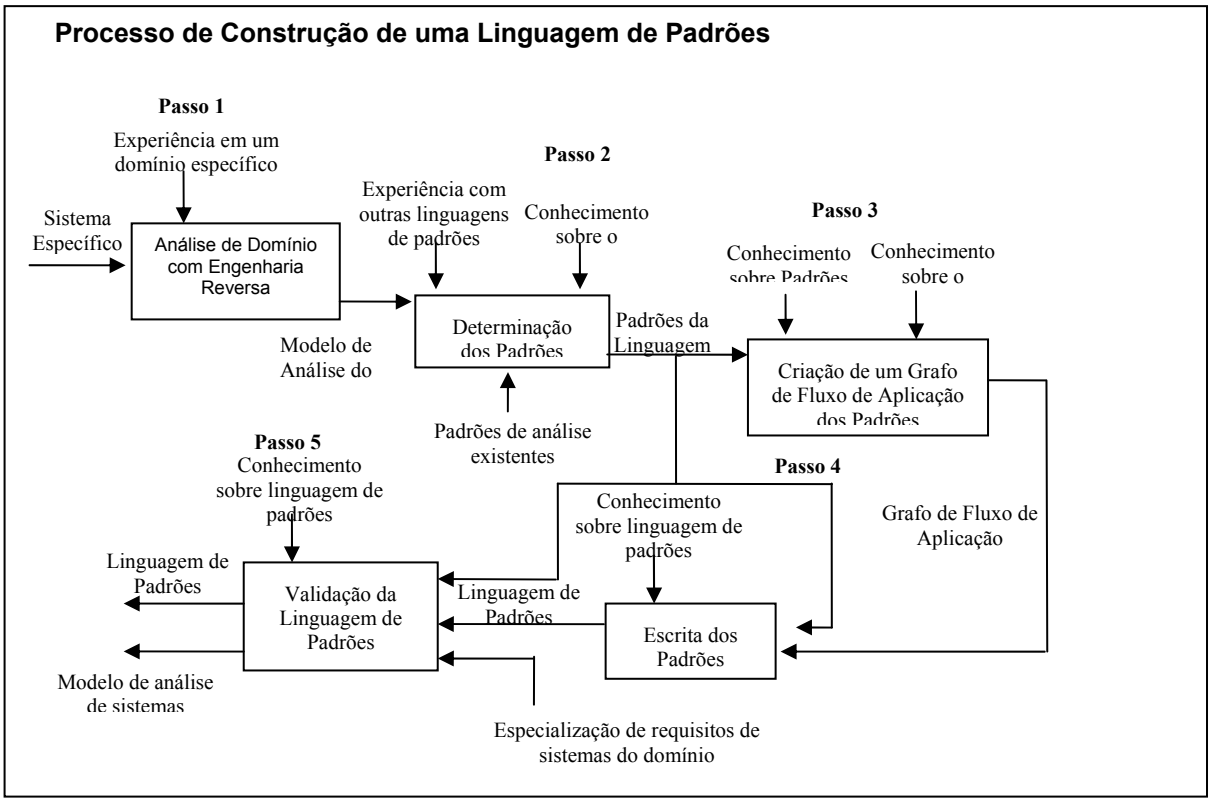


Figura 3. Processo de construção de uma linguagem de padrões com base em análise de domínio [10]

O passo 1, *Análise de Domínio com Engenharia Reversa*, foi realizado com base na engenharia reversa de sistemas legados. Roberts e Johnson [13] sugerem que o desenvolvimento de um *framework* deve acontecer após o desenvolvimento de três aplicações concretas que irão fornecer conhecimento suficiente para permitir de forma

gradativa a generalização dessas aplicações. Embora o objetivo deste passo não seja a elaboração de um *framework*, esses conceitos podem ser aplicados para a análise de domínio.Os sistemas escolhidos são sistemas reais que atualmente são utilizados pelas Clínicas de Reabilitação Física Dom Bosco, das Faculdades Salesianas de Lins - SP. O processo usado para engenharia reversa desses sistemas é específico, pois são desenvolvidos em ambiente ZIM [11]. Dois apoios computacionais, `Gera Java` e `Gera SQL`, são utilizados para facilitar a recuperação desses sistemas, a partir das informações existentes no dicionário de dados das aplicações, gerando arquivos com finalidades específicas. `Gera Java` lê as definições do dicionário de dados e gera um arquivo, chamado `Classes.java`, considerando cada tabela ZIM, do sistema legado, como uma classe. `Gera SQL` lê as definições do dicionário de dados e gera três arquivos (`fldsdocs, relsdoc, Tabelas`) que facilitam a identificação das tabelas para o novo sistema. O processo de engenharia reversa pode ser apoiado por uma ferramenta CASE, como *Rational Rose*[12], que disponibilize o recurso de conversão de classes Java para um diagrama de classes. Maiores detalhes sobre o processo de engenharia reversa podem ser encontrados em [7].

O Modelo de Domínio é elaborado buscando-se a similaridade existente entre as classes de cada aplicação. Assim, tem-se o modelo do domínio exibido na Figura 4, com letras inseridas em cada classe para representar as similaridades observadas entre os três sistemas: S – classes Similares existentes nos três sistemas; PS – classes Parcialmente Similares existentes em dois dos três sistemas e E – classes Exclusivas que existem somente em um dos sistemas.

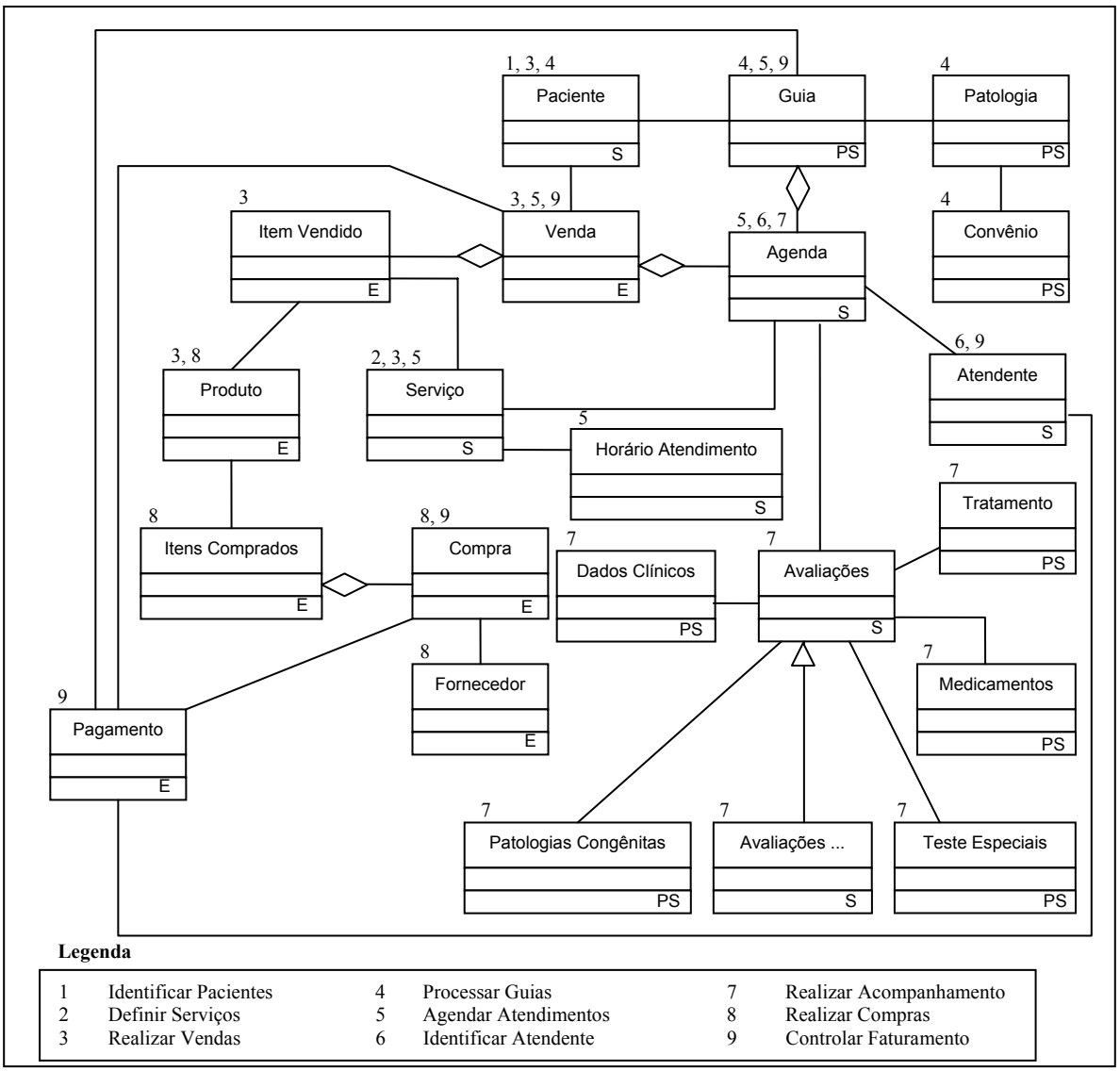


Figura 4. Modelo de classes do domínio para sistemas de gerenciamento de clínicas de reabilitação(SiGCLI).

O passo 2, *Determinação dos Padrões*, agrupa as classes do modelo do domínio, de acordo com suas funcionalidades, formando um conjunto de classes que determinará os futuros padrões da linguagem. Analisando-se o funcionamento dos sistemas escolhidos, verificou-se quais classes são necessárias para tratar de funcionalidades específicas agrupando e identificando-as por um nome, que define o padrão. Na Figura 4, toda classe possui uma numeração, no canto superior esquerdo, que identifica a qual padrão ela pertence. As classes pertencentes a mais de

um padrão, possuem números separados por vírgulas. A Legenda apresenta o nome dos padrões com seus respectivos números de identificação. Por exemplo, *Realizar Vendas* (3), é formado pelas classes *Paciente*, *Produto, Serviço, Venda, Item Vendido.*

O passo 3, *Criação de um Grafo de Fluxo de Aplicação dos Padrões*, prevê a definição da ordem de aplicação e interação entre os padrões da SiGCli, Figura 5. Uma das formas para a elaboração do grafo é iniciar pelos padrões que representam as funcionalidades mais básicas do domínio. Três grupos são formados: *Informações Básicas*, *Atendimentos* e *Controle de Transações Financeiras*, com os padrões extraídos do diagrama de classes apresentado na Figura 4 e especificados na Legenda.

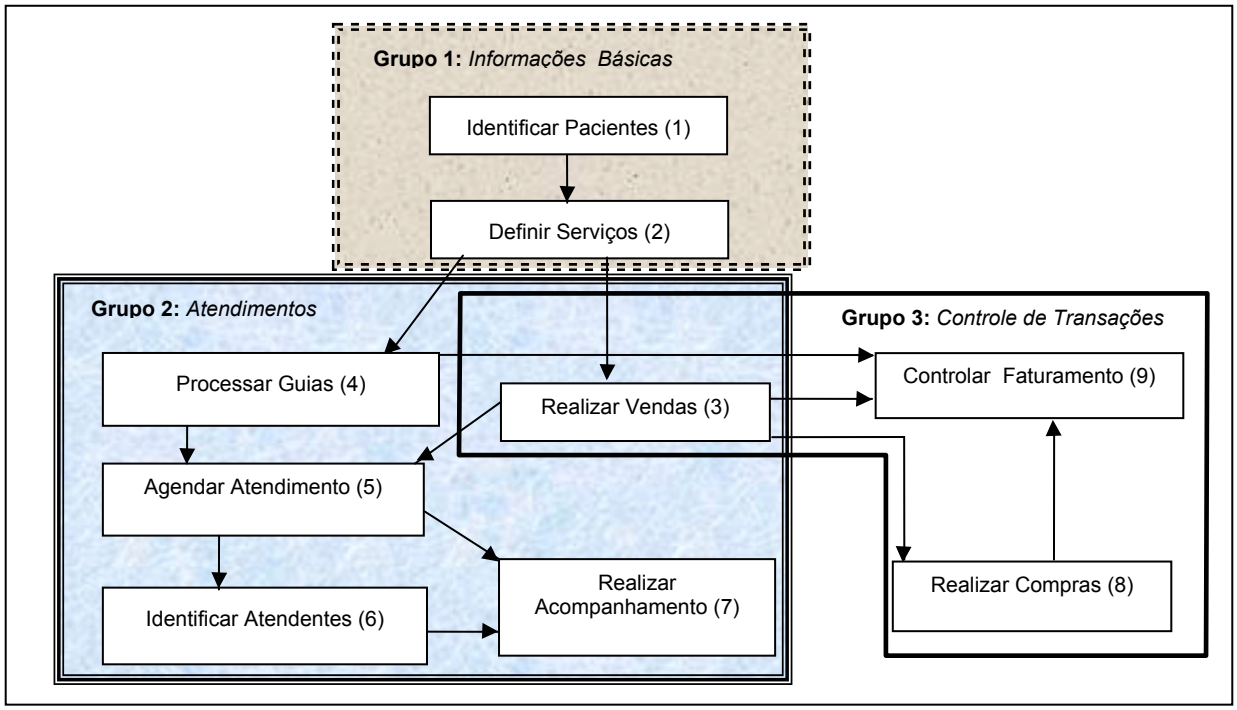


Figura 5. Grafo do fluxo de aplicação dos padrões da SiGCLI

O passo 4, *Escrita dos Padrões,* é o passo mais importante, uma vez que o padrão deve ser bem elaborado para que não haja dúvidas quanto à sua interpretação pelos analistas que o utilizarão. Existem alguns formatos propostos na literatura, sendo que o adotado foi o seguinte: "nome", "contexto", "problemas", "estrutura", "participantes", "padrões relacionados". O passo 5, *Validação da Linguagem de Padrões,* objetiva aplicar os padrões a diferentes sistemas do domínio estudado, comprovando sua eficiência.

Durante a realização dos passos 4 e 5 observou-se que o domínio da SiGCli apresenta pontos comuns com o domínio da GRN, assim, procedeu-se um mapeamento entre os padrões das duas linguagens. O domínio da SiGCLi possui algumas funcionalidades específicas que diferem das aplicações do domínio da GRN. Dessa forma, algumas adaptações nos padrões da GRN são necessárias, caracterizando a SiGCli como um sub-domínio da GRN.

A Tabela 1 exibe o mapeamento entre os padrões da SiGCli e os da GRN quando aplicados a sistemas de clínicas de reabilitação. O padrão 7 da SiGCli não possui mapeamento direto na GRN. Sua funcionalidade é especifica para o domínio de clínicas de reabilitação devido à necessidade de acompanhamento de todas as sessões realizadas por um paciente durante o seu tratamento nas clinicas de reabilitação.

Tabela 1- Mapeamento entre os padrões da SiGCli com os da GRN com foco no tratamento do Paciente.

SiGCli		**GRN**	
Nº Padrão	Padrão	Nº Padrão	Padrão
1	Identificar Pacientes	1	Identificar o Recurso
		2	Quantificar o Recurso
2	Definir Serviços	1	Identificar o Recurso
		2	Quantificar o Recurso
3	Realizar Vendas	1	Identificar o Recurso
		2	Quantificar o Recurso
		6	Comercializar o Recurso
		11	Itemizar Transação do Recurso
4	Processar Guias	9	Manter o Recurso

5	Agendar Atendimentos	14	Identificar Tarefas de Manutenção
6	Identificar Atendentes	13	Identificar o Executor da Transação
7	Realizar Acompanhamento		*Não possui mapeamento*
8	Realizar Compras	6	Comercializar o Recurso
		11	Itemizar Transação do Recurso
9	Controlar Faturamento	12	Pagar pela Transação do Recurso

O mapeamento ilustrado na Tabela 1 foi elaborado com o foco no paciente que procura uma clínica para se reabilitar de uma lesão. Nesse caso, tem-se que um recurso, o paciente, deve sofrer a manutenção, tratamento, realizada pela clínica. A GRN possibilita um outro foco com base nos serviços oferecidos pela clínica. Nessa situação tem-se que um recurso, o serviço oferecido pela clínica, é alugado para um determinado paciente para realizar o tratamento. Esse mapeamento é o ilustrado na Tabela 2. Como se pode observar, analisando as duas tabelas, apenas os padrões 3 e 4 da SiGCli sofreram alterações com relação à aplicação padrões da GRN. Usando as informações dessas tabelas a instanciação das aplicações com o *framework* GREN é facilitada, uma vez que os padrões já foram mapeados cabendo ao engenheiro de software somente aplicá-los conforme a definição do GREN-Wizard.

Tabela 2- Mapeamento entre os padrões da SiGCli com os da GRN com foco no aluguel de serviços da clínica.

SiGCli		**GRN**	
Nº Padrão	Padrão	Nº Padrão	Padrão
1	Identificar Pacientes	1	Identificar o Recurso
		2	Quantificar o Recurso
2	Definir Serviços	1	Identificar o Recurso
		2	Quantificar o Recurso
3	Realizar Vendas	1	Identificar o Recurso
		2	Quantificar o Recurso
		6	Comercializar o Recurso
		11	Itemizar Transação do Recurso
4	Processar Guias	5	Reservar o Recurso
5	Agendar Atendimentos	4	Locar o Recurso
6	Identificar Atendentes	13	Identificar o Executor da Transação
7	Realizar Acompanhamento		*Não possui mapeamento*
8	Realizar Compras	6	Comercializar o Recurso
		11	Itemizar Transação do Recurso
9	Controlar Faturamento	12	Pagar pela Transação do Recurso

4. Estudo de Caso: Utilização do *Framework* GREN para Instanciar Aplicações no Domínio SiGCli.

Esta seção apresenta um estudo de caso exemplificando o uso do *framework* GREN para instanciação de uma Clínica de Fisioterapia. Essa clínica controla todos os atendimentos realizados em seus pacientes e o seu objetivo principal é armazenar os acompanhamentos e avaliações realizadas pelos atendentes. Esses atendentes são alunos do curso de Fisioterapia e devem ser avaliados na disciplina de estágio supervisionado, existente em sua grade curricular. Esses atendentes não são remunerados e o controle do faturamento da clínica também não é de interesse nesse sistema. O modelo de classes do sistema é apresentado na Figura 6, sendo resultado da aplicação dos padrões (1) *Identificar Pacientes*, (2) *Definir Serviços*, (4) *Processar Guias*, (5) *Agendar Atendimentos*, (6) *Identificar Atendentes* e (7) *Realizar Acompanhamento* da linguagem de padrões SiGCli.

A instanciação da aplicação no *framework* GREN segue o grafo de fluxo de aplicação dos padrões da GRN que é diferente do da SiGCli. Logo, os padrões sem correspondência nessas duas linguagens devem ser instanciados manualmente pelo desenvolvedor, o que nem sempre é uma tarefa trivial.

A Tabela 3 define o mapeamento entre as classes disponíveis no *framework* GREN e as classes existentes na linguagem de padrões SiGCli. Somente o padrão 7 da SiGCli, *Realizar Acompanhamento*, não possui correspondência para a instanciação usando o *framework*. Cada classe no GREN tem um papel definido quando se pretende instanciar uma aplicação do domínio SiGCli. Dessa forma, para instanciar a aplicação apresentada pela

Figura 6, têm-se os padrões da SiGCli aplicados na seguinte ordem (1), (4), (6), (5), (2), (7) de acordo com o grafo de fluxo de aplicação dos padrões da GRN.

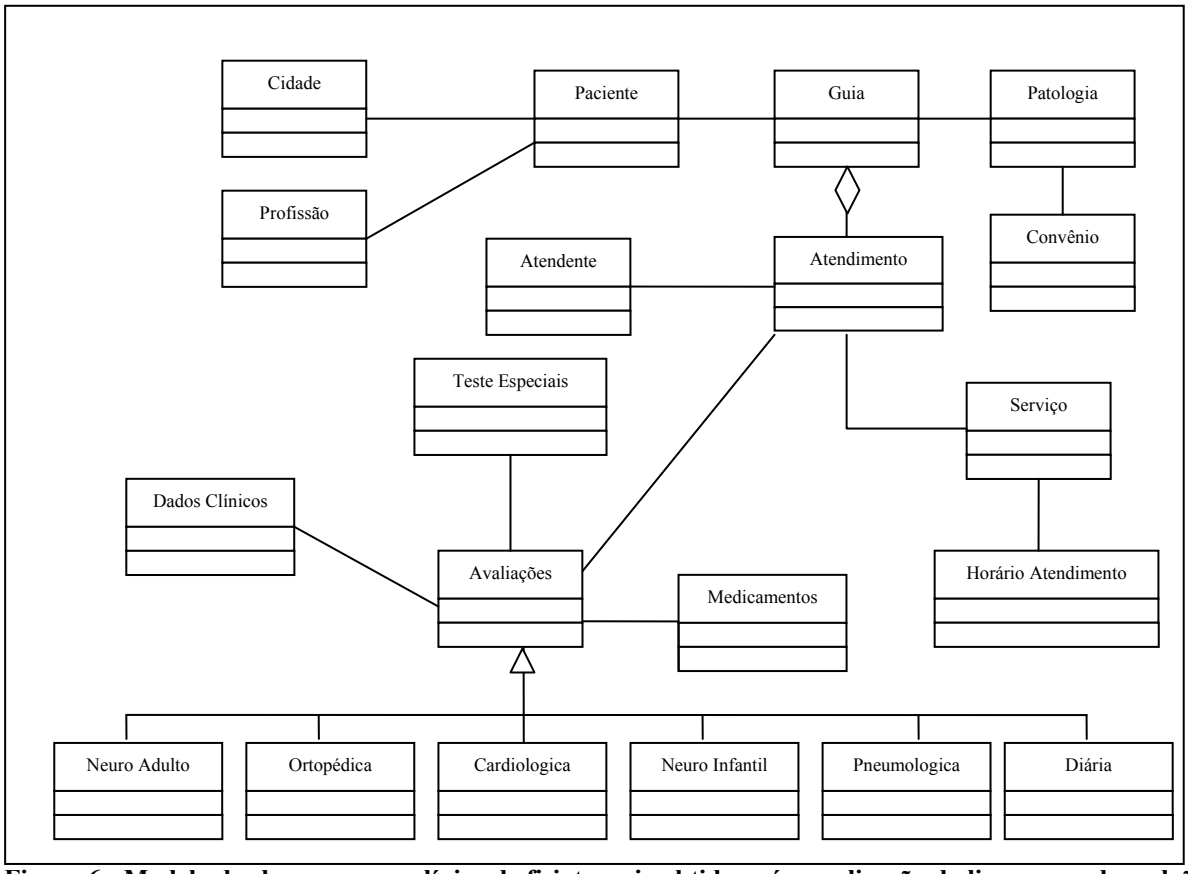


Figura 6 - Modelo de classes para a clínica de fisioterapia obtido após a aplicação da linguagem de padrões SiGCli.

Tabela 3 - Mapeamento entre as classes apresetadas no GREN-Wizard com as classes da SiGCli

SiGCli			**GREN**			
Nº	***Padrão***	***Classes***	***Nº***	***Padrão***	***Variante***	***Classes***
1	Identificar Pacientes	Paciente	1	Identificar o Recurso	Múltiplos tipos de recursos	Recurso
		Informação Adicional	2	Quantificar o Recurso	Recurso Simples	Tipos de Recurso
2	Definir Serviços	Serviço	1	Identificar o Recurso	Default	Recurso
		Tipo do Serviço	2	Quantificar o Recurso	Recurso Simples	Tipos de Recurso
3	Realizar Vendas	Produto	1	Identificar o Recurso e	Default	Recurso
		Unidade de Medida	2	Quantificar o Recurso	Recurso Mensurável	Unidade de Medida
		Paciente	6	Comercializar o Recurso	Venda sem origem	Destino
		Venda				Comercialização do Recurso
		Serviço / Produto				Recurso
		Venda	11	Itemizar Transação do Recurso	Default	Transação do Recurso
		Item Vendido				Item de transação
		Serviço / Produto				Recurso
4	Processar Guias	Paciente	9	Manter Recurso	Default	Recurso

		Guia				Manutenção do Recurso
		Convênio				Origem
5	Agendar Atendimentos	Venda	14	Identificar Tarefas de Manutenção	Default	Comercialização do Recurso
		Guia				Manutenção do Recurso
		Agenda				Tarefa de Manutenção
6	Identificar Atendentes	Agenda	13	Identificar Executor da Transação	Default	Tarefa de Manutenção
		Atendente				Executor da Tarefa
7	Realizar Acompanhamento					
8	Realizar Compras	Fornecedor	6	Comercializar o Recurso	Default	Origem
		Compra				Comercialização do Recurso
		Produto				Recurso
		Compra	11	Itemizar Transação do Recurso	Default	Transação do Recurso
		Item Comprado				Item de transação
		Produto				Recurso
9	Controlar Faturamento	Venda	12	Pagar pela Transação do Recurso		Transação do Recurso
		Guia				Transação do Recurso
		Compra				Transação do Recurso
		Pagamento				Pagamento
		Taxa de Juros				Taxa de Juros
		Taxa de Multa				Taxa de Multa

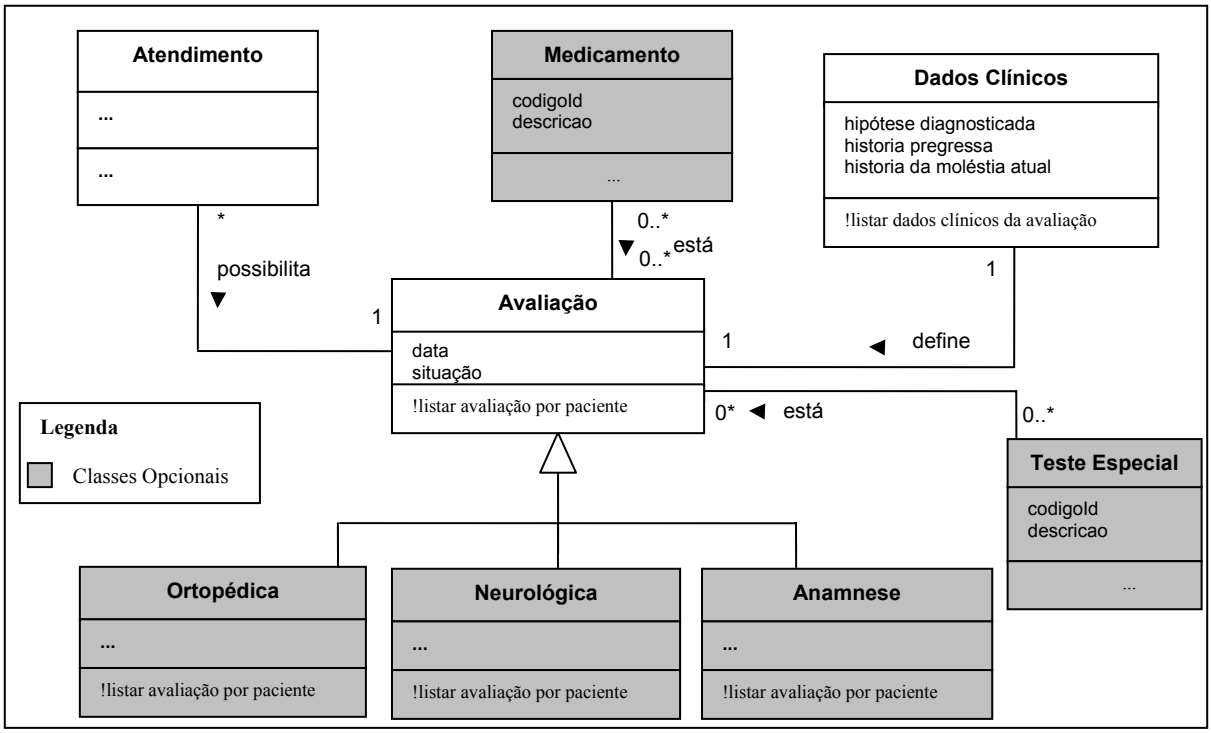


Figura 7 – Diagrama de classes do padrão 7 Realizar Acompanhamento da SiGCli.

Esse mapeamento, Tabela 3, facilita o uso do GREN-Wizard para instanciar aplicações no domínio SiGCli, uma vez que o desenvolvedor consegue identificar, de forma rápida, qual papel cada classe da SiGCli exerce em relação a sua possível aplicação no GREN-Wizard. A Figura 8 exibe a aplicação, no GREN-Wizard, do padrão 2 da SiGCli *Definir Serviços, que* é mapeado no padrão *Identificar Recurso*, da GRN, com a variação *Default,* como é definido na Tabela 3. A instanciação do padrão 2 da SiGCli acontece sem problemas, uma vez que todas as classes possuem mapeamento com as classes definidas para o Identificar Recursos da GRN.

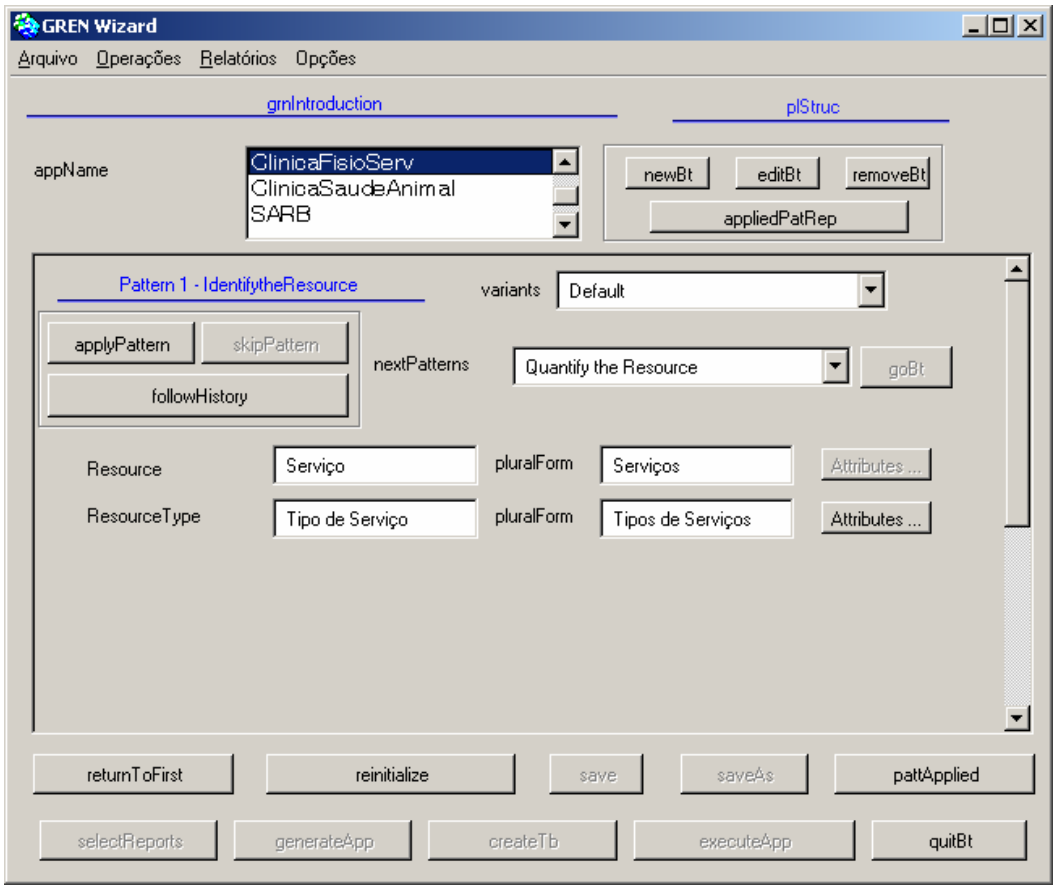


Figura 8 - Aplicando o padrão 2 da SiGCli, *Definir Serviços*, no GREN-Wizard

As classes que não possuem mapeamento devem ser instanciadas através de um processo manual, sem a ajuda do Wizard, na qual deve-se investigar se existem classes no *framework* GREN que podem servir como *template* para as novas classes. Para essa implementação o desenvolvedor deve possuir conhecimentos sobre programação em linguagem *Smalltalk* e estudar a estrutura interna do *framework* para que essas classes possam utilizar os recursos disponíveis pelo *framework*. Todas as adaptações sofridas pelo framework devem ser documentadas para um futuro aprimoramento da linguagem de padrões GRN e do GREN-Wizard. A Figura 7 apresenta o diagrama de classes do padrão 7 da SiGCli, *Realizar Acompanhamentos*, que não é coberto pelo *framework.*

5. Considerações Finais

O GREN-Wizard é uma ferramenta desenvolvida em *Smalltalk* que foi elaborada para facilitar a instanciação de aplicações que venham a usar o *framework* GREN e são cobertas pelo domínio da linguagem de padrões GRN. Ela possui uma interface amigável que guia o desenvolvedor durante o processo de instanciação da aplicação. A única exigência da ferramenta é que o desenvolvedor conheça a GRN e saiba quais padrões devem ser aplicados para obter a aplicação desejada.

Durante a elaboração da SiGCli, procurou-se sempre buscar padrões associados aos da GRN, para facilitar uma futura instanciação usando o GREN-Wizard. Entretanto, algumas funcionalidades específicas da SiGCli não são cobertas por esses padrões. Dessa forma, foram feitas adaptações aos padrões da GRN sempre que necessário. Após a elaboração dos padrões que definem a SiGCli, foram realizados três estudos de caso com o objetivo de minimizar os esforços na instanciação das aplicações. Para isso, os três sistemas usados no processo de análise de domínio foram remodelados, seguindo os padrões propostos pela SiGCli e foram aplicados no GREN-Wizard.

Durante o processo de instanciação, usando o GREN-Wizard, foram identificados dois fatores importantes que dificultam a instanciação de aplicações: a) a definição dos papeis que cada classe exerce nos padrões mapeados. Por exemplo, a classe *Paciente* assume dois papéis distintos durante a instanciação da aplicação, ora sendo o recurso ora sendo o destino da comercialização de um produto, outras classes também sofrem esse tipo de mutação. Tais mudanças de papéis não são cobertas pelo *framework* GREN, o que dificulta o entendimento do desenvolvedor para instanciar a aplicação; b) as funcionalidades específicas do domínio da SiGCli que não são previstas pelo *framework* exigindo do desenvolvedor conhecimento profundo do *framework* GREN e da linguagem *SmallTalk* para implementá-los.

Sendo o interesse por sistemas implementados em linguagem Java com interface Web usando a arquitetura três camadas e *Smalltalk* uma linguagem bem difundida no meio científico, mas sem grande representatividade no mercado, os esforços para a alteração do *framework* GREN são grandes e os resultados obtidos não seriam satisfatórios as necessidades reais das aplicações. Um novo padrão deveria ser incorporado à linguagem de padrões GRN, o padrão 7 da SiGCli, e implementado em *Smalltalk* para ser adicionado à estrutura do *framework* de modo que essa arquitetura não fosse violada. Dessa forma, alterações seriam necessárias nas camadas: *GREN-Negócio*, *GREN-Interface Gráfica com o Usuário* e *GREN-Wizard* do *framework* GREN.

Devido a não viabilidade da utilização do *framework* GREN com a linguagem de padrões SiGCli optou-se pelo desenvolvimento de um gerador de aplicações específico para esse domínio, apoiado nessa linguagem de padrões, produzindo aplicações Java com interface Web em arquitetura três camadas [7].

Referências

[1] BRAGA, R. T. V.; MASIERO, P. C. A process for framework construction based on a pattern language. In: *26th Annual International Computer Software and Applications Conference (COMPSAC 2002)*, Oxford–England, to appear, 2002.

[2] RÉ, R. *Um processo para construção de frameworks a partir da engenharia reversa de sistemas de informação baseados na Web: Aplicação ao domínio dos leilões virtuais*. Dissertação de Mestrado, ICMC/USP, São Carlos – SP, 2002.

[3] FAYAD, M. E.; E.JOHNSON, R.; SCHMIDT, D. C. *Building application frameworks: Objectoriented foundations of framework design*. John Wiley & Sons, 1999.

[4] COPLIEN, J. O. *Software design patterns: Common questions and answers* in L. Rising – The Patterns Handbook: Techniques, Strategies, and Applications, Cambridge University Press, p. 311–320, 1998.

[5] JOHNSON, R. E. Frameworks = (components + patterns). *Communications of the ACM*, v. 40, n. 10, p. 39–42, 1997b.

[6] JOHNSON, R. E.; RUSSO, V. *Reusing object-oriented designs*. Rel. T´ec. UIUCDCS 91-1696, University of Illinois, 1991.

[7] PAZIN, A. Um Gerador de Aplicações para o Domínio de Clínicas de Reabilitação. Dissertação de Mestrado a ser apresentada – Programa de Pós Graduação em Ciência da Computação – Universidade Federal de São Carlos, São Carlos – SP, agosto/2004

[8] BRAGA, R. T. V.; GERMANO, F. S. R.; MASIERO, P. C. A family of patterns for business resource management. In: *5th Annual Conference on Pattern Languages of Programs (PLOP'98)*, Washington University in St. Louis – Missouri, USA, on-Line. Disponível em http://jerry.cs.uiuc.edu/~plop/plop98/final_submissions. Consultado em 31/01/2003., 1998.

[9] BRAGA, R. T. V.; GERMANO, F. S. R.; MASIERO, P. C. A pattern language for business resource management. In: *6th Pattern Languages of Programs Conference (PLoP'99)*, Monticello – IL, USA, 1999.

[10] BRAGA, R. T. V. Um Processo para a Construção e Instanciação de *Frameworks* baseado em uma Linguagem de Padrões para um Domínio Específico. Tese de Doutorado, ICMC/USP, São Carlos-SP, 2003.

[11] BROWN, A. F. Como desenvolver aplicações com o SGBD ZIM. Rio de Janeiro: LTC – Livros Técnicos e Científicos Editora, 1990. 297p.

[12] ROSE - Visual Modeling with Rational Rose Home. Página com mais informações disponível na URL:http://www.rational.com/products/rose/index.jsp. Último acesso em 16 de Abril de 2003.

[13] ROBERTS, D.; JOHNSON, R. *Evolving frameworks: A pattern language for developing object oriented frameworks* in Martin, R.C., Riehle, D. , Buschmann, F. Pattern Languages of Program Design 3, Addison-Wesley, p. 471–486, 1998.

Um Meta-modelo para o Processo de Sistemas com RV: Perspectiva da Qualidade no Uso Provida por Princípios da IHC

Milena Marquezin Olher
Júnia Coutinho Anacleto Silva
DC - UFSCar
C. P. 676 - CEP 13565-905 - São Carlos - SP
(016) 260-8233 / 8618
{milena, junia}@dc.ufscar.br

Abstract

Software system quality can be considered under three perspectives: internal quality, external quality and quality in use. Considering the perspective of the quality in use, this paper analyzes the characteristics of software systems with Virtual Reality and of their process, mainly about the human-computer interaction style that the use of Virtual Reality technology want to provide and, starting from the discussion of the specific methodologies for these software systems, this article proposes a meta-model for the process of software systems with Virtual Reality that contemplates not only the perspectives of the internal quality and of the external quality, but also the perspective of the quality in use through Human-Computer Interaction principles, adopting and incorporating focus on the user, iterations of design and usability criteria for theses software systems through the conception, design and implementation of software systems with Virtual Reality.
Keywords: Virtual Reality, Human-Computer Interaction, Quality in Use, Process of Software, Usability, Usability Criteria, Focus on the User e Iterations of Design.

Resumo

Qualidade de sistema de software pode ser considerada sob três perspectivas: qualidade interna, qualidade externa e qualidade no uso. Considerando a perspectiva da qualidade no uso, este artigo analisa as especificidades de sistemas de software com Realidade Virtual e as implicações no seu processo, principalmente no que concerne ao estilo de interação humano-computador que o emprego da tecnologia de Realidade Virtual objetiva prover e, a partir da apresentação e discussão das metodologias para esses sistemas de software, este artigo propõe um meta-modelo para o processo de sistemas de software com Realidade Virtual que contempla não apenas as perspectivas da qualidade interna e da qualidade externa, mas também a perspectiva da qualidade no uso através de princípios da Interação Humano-Computador, adotando e incorporando enfoque no usuário, iteratividade de projeto e critérios de usabilidade específicos para esses sistemas de software ao longo da concepção, projeto e implementação de sistemas de software com Realidade Virtual.
Palavras-chave: Realidade Virtual, Interação Humano-Computador, Qualidade no Uso, Processo de sistemas, Usabilidade, Critérios de Usabilidade, Enfoque no Usuário e Iteratividade de Projeto.

1. INTRODUÇÃO

O modelo de qualidade desenvolvido pela ISO/IEC 9126-1 apresenta três diferentes perspectivas para a qualidade de sistema de software (sistema) [2]: 1) qualidade interna, mensurada através das propriedades estáticas do código do sistema, 2) qualidade externa, medida através das propriedades dinâmicas do código do sistema quando executado e 3) qualidade no uso, mensurada através da extensão na qual o sistema atende as necessidades do usuário em um ambiente de uso. A perspectiva da qualidade no uso é a eficiência, produtividade e satisfação do usuário ao realizar tarefas representativas em um ambiente de uso realístico, ou seja, é a visão da qualidade do sistema a partir do ponto de vista do usuário. O alcance a essa perspectiva pode ser considerado como um dos maiores objetivos no processo de sistemas interativo, pois visa que o sistema desenvolvido possa ser usado para o propósito pretendido [1][2].

Nesse contexto, sistema de software com interface com o usuário baseada na tecnologia de Realidade Virtual (sistema com RV), a qual propõe que o processo de interação entre o sistema e seus usuários finais (usuários) ocorra de forma mais natural e intuitiva aos usuários, apresenta características mais específicas [9] [17] [22] que podem afetar de modo considerável a qualidade no uso desses sistemas por seus usuários, aumentando a complexidade do processo de sistemas com RV e devendo ser consideradas também sob essa perspectiva da qualidade ao longo da concepção, projeto e implementação desses sistemas.

Dessa forma, este trabalho analisa as especificidades de sistemas com RV e as implicações no seu processo, e, a partir da apresentação e discussão das metodologias para esses sistemas, propõe um meta-modelo para o processo de sistemas RV que contemple não apenas as perspectivas da qualidade interna e da qualidade externa, mas também a perspectiva da qualidade no uso através de princípios da Interação Humano-Computador (IHC), adotando e incorporando critérios de usabilidade específicos para esses sistemas, enfoque no usuário e iteratividade de projeto ao longo da concepção, projeto e implementação de sistemas RV, a fim de garantir o estilo de interação humano-computador que o emprego da tecnologia de RV objetiva prover.

Na segunda seção deste artigo são apresentadas as especificidades de sistemas com RV e as implicações no seu processo considerando a qualidade no uso. A terceira seção discute metodologias para sistemas com RV. A quarta seção compara as metodologias para sistema com RV. Na quinta deste artigo, como resultado deste trabalho, é proposto um meta-modelo para o processo de sistemas com RV, sendo realizada a incorporação de enfoque no usuário, iteratividade de projeto e critérios de usabilidade. A sexta seção mostra as conclusões.

2. ESPECIFICIDADES DE SISTEMAS COM RV E IMPLICAÇÕES NO PROCESSO DE SISTEMAS COM RV CONSIDERANDO A QUALIDADE NO USO

Segundo Tanriverdi [22], considera-se que sistema com RV é composto por (a) usuário e ou outras entidades externas, manipulam e trocam informação com o sistema através da interface, (b) interface, consiste de objetos de interface (entidades que possuem regras e identidades bem definidas) e dados (entradas recebidas de usuários e ou de outras entidades externas), (c) controle de diálogo, permite a comunicação entre interface e aplicação e (d) aplicação, contém regras e conhecimento que definem a lógica do sistema com RV.

Sistema com RV possibilita que o processo de interação ocorra de forma mais inerente aos seres humanos, uma vez que é realizado no plano tridimensional e de forma mais diversificada quanto aos sentidos humanos.

Considerando os recursos de interface para prover o processo de interação proposto, sistema com RV exibe diferentes características se comparado com sistema convencional. Enquanto interfaces de sistema convencionais (interfaces convencionais) apresentam principalmente exibições 2D, interfaces de sistema com RV (interfaces com RV) apresentam exibições 2D e, principalmente, 3D. Interfaces convencionais contêm tipicamente objetos virtuais gerados por computador, ao passo que interfaces com RV contêm objetos físicos e virtuais que podem coexistir trocando informações entre si. Quanto às suas características comportamentais, enquanto objetos de interfaces convencionais normalmente exibem comportamentos predeterminados, objetos de interfaces com RV exibem comportamentos autônomos e comunicações entre si que podem mudar seus estados e comportamentos. Enquanto interfaces convencionais suportam interações explícitas (como comandos do usuário usando teclado), interfaces com RV suportam interações explícitas e, principalmente, implícitas (como movimentos das mãos do usuário usando luvas) [22]. Interfaces convencionais provêem interações discretas entre usuário e sistema por meio de dispositivos seriais como teclado ou mouse, ao passo que interfaces com RV provêem interações contínuas entre usuário e sistema por meio de dispositivos paralelos como capacete e luvas [9]. Considerando o desempenho para prover o processo de interação proposto, sistema com RV deve prover desempenho em tempo real, garantindo níveis aceitáveis de imersão, interação e navegação [17].

Nesse contexto, produtos com excelência técnica não são o bastante, isto é, produtos também devem ser de fácil uso e ajuste às práticas e atividades de trabalho do consumidor ou usuário [2]. Nesse contexto, algumas tentativas têm sido realizadas para ampliar a percepção de qualidade. No modelo da ISO/IEC 9126, qualidade é caracterizada do ponto de

vista do usuário como: funcionalidade, confiabilidade, eficiência, manutenibilidade, portabilidade e ainda, usabilidade. Usabilidade é definida na ISO/IEC 9126 como um conjunto de atributos que afetam o esforço necessário para o uso, e a avaliação individual de tal uso, por um conjunto de usuários declarados ou implícitos. Porém, embora desenvolvedores de sistema queiram saber quais atributos incorporar ao código fonte para reduzir o esforço necessário para o uso, presença ou ausência de atributos predefinidos no código pode não assegurar usabilidade [2]. No novo modelo incluído na revisão da ISO/IEC 9126, qualidade é caracterizada através de três diferentes perspectivas: qualidade interna, qualidade externa e qualidade no uso. Qualidade no uso é definida como a extensão na qual o sistema atende as necessidades do usuário em um ambiente de uso. Usabilidade é definida na revisão da ISO/IEC 9126 como a capacidade do sistema ser entendido, apreendido, usado e satisfatório para o usuário, quando usado sob condições especificadas. Os benefícios dessa perspectiva mais abrangente de qualidade incluem maior eficácia, melhor produtividade e aceitação e redução de erros e de treinamentos [2].

De acordo com Smith [18], o processo de sistemas com RV e seus desenvolvedores devem estar focados na utilização da tecnologia de RV de tal modo que os sistemas desenvolvidos atendam aos anseios de seus usuários e sigam critérios de qualidade. Porém, desenvolver sistemas com RV que atendam valor estético, funcional e de uso esperado por seus usuários se apresenta como desafio no emprego dessa tecnologia, pois ainda precisam ser ponderadas inúmeras questões, dentre elas melhor analisar e formalizar o processo desses sistemas, considerando a perspectiva da qualidade no uso, a qual é focada por este trabalho através de enfoque no usuário, de iteratividade de projeto e de critérios de usabilidade para sistema com RV. Nota-se que não somente ferramentas e abordagens detalhadas para o processo de sistemas com RV dificilmente são encontradas ou amplamente aceitas, mas também abordagens detalhadas para o processo de sistemas com RV [16] [17] [18] [22] [23] que visem critérios de usabilidade para esses sistemas distribuídos ao longo de sua concepção, projeto e implementação.

Dessa forma, acredita-se que o processo de sistemas com RV deva estar pautado por metodologias que considerem as perspectivas de qualidade, mais especificamente princípios IHC no que diz respeito ao estilo de interação que o emprego da tecnologia de RV objetiva prover em sistemas interativos, garantindo à perspectiva da qualidade no uso através da ponderação das implicações exigidas ao longo da concepção, projeto e implementação de sistemas com RV.

2.1 Enfoque no Usuário no Contexto de RV

O processo de sistemas com RV, bem como os desenvolvedores desses sistemas devem estar voltados para a utilização da tecnologia de RV de tal modo que os sistemas desenvolvidos apoiem aos requisitos de seus usuários atendendo critérios de qualidade [2] no que diz respeito ao seu valor estético, funcional e de uso. Contudo, artigos ou pesquisas que detalhem o processo de sistemas com RV, de tal forma que os benefícios do emprego dessa tecnologia em sistemas interativos sejam explorados, dificilmente são encontrados [18]. Observa-se que, segundo este trabalho, analisar e formalizar amplamente o processo de sistemas com RV, em relação à sua abrangência não somente na área de ES, mas ainda na área da IHC, se apresenta como importante desafio para melhor empregar essa tecnologia em sistemas interativos. Segundo Scaife [16], as características de sistemas com RV, principalmente no que se refere à alta interação entre esses sistemas e seus usuários, têm levado à utilização de metodologias para sistemas com RV baseadas em abordagens da IHC [4] como o Projeto Centrado no Usuário (PCU) e o Design Participativo (DP), as quais estão fundamentadas no enfoque no usuário durante o processo de projeto [6]. A importância da atividade de prototipação, a qual permite que avaliações iterativas de um sistema sejam realizadas pelos usuários, é reconhecida na abordagem PCU, e portanto também na abordagem DP, uma vez que é considerada como um modo específico do PCU. A utilização do PCU e, segundo este trabalho, do DP, é realizada em conjunto com o modelo de processo de Prototipação [14] da Engenharia de Software (ES), pois esse modelo de processo se adapta às exigências dessas abordagens. Dessa forma, considera-se que o PCU e DP estão baseados no modelo de processo de Prototipacão da ES.

Desse modo, devido à alta integração entre sistemas com RV e seus usuários, acredita-se que abordagens com enfoque no usuário se tornam requisito essencial para que esses sistemas atendam a padrões de usabilidade esperados [18] e, desse modo, devem estar inseridas no processo desses sistemas.

2.2 Iteratividade de Projeto no Processo de sistemas com RV

Segundo Pressman [14], as fases genéricas da ES são encontradas em todos os processos de sistemas, independentemente do modelo de processo seguido e da área de aplicação, tamanho ou complexidade do sistema. Segundo Smith [18], o processo de sistemas com RV é composto pelas etapas de especificação, implementação e avaliação, exigindo iteratividade de projeto [17][18][21][22].

Vale notar que é possível correlacionar as fases genéricas da ES e as etapas do processo de sistemas com RV com base em suas atividades. A etapa de especificação corresponde à fase genérica de definição da ES. As etapas de implementação e avaliação correspondem à fase genérica de desenvolvimento da ES. Nota-se que, de acordo com o

escopo deste trabalho, não há etapas correspondentes à fase genérica de manutenção da ES para contemplar o processo de sistemas com RV, pois nessa fase são reaplicados os passos das fases de definição e desenvolvimento.
Dessa forma, acredita-se que metodologias para sistemas com RV devam apresentar, ao menos, etapas iterativas correspondentes às fases genéricas de definição e desenvolvimento da ES, bem como suas atividades, independente da nomeação dada por cada metodologia a essas fases e atividades, uma vez que essas fases são encontradas em todos os processos de sistemas e iteratividade de projeto é exigida para o processo de sistemas com RV.

2.3 Usabilidade no Contexto de RV

De acordo com Smith [18], o processo de sistemas com RV e seus desenvolvedores devem estar focados na utilização da tecnologia de RV de tal modo que o sistema desenvolvido atenda aos anseios de seus usuários e siga critérios de qualidade como usabilidade. Porém, desenvolver sistema com RV que apresente valor estético e funcional e atenda aos anseios de seus usuários em relação à sua usabilidade, se apresenta como desafio na utilização dessa tecnologia, pois ainda precisam ser ponderadas inúmeras questões, dentre elas melhor analisar e formalizar o processo desses sistemas, considerando a perspectiva da qualidade no uso, a qual é focada por este trabalho através de enfoque no usuário, de iteratividade de projeto e de critérios de usabilidade para sistema com RV. Nota-se que não somente ferramentas e abordagens detalhadas para o processo de sistemas com RV dificilmente são encontradas ou amplamente aceitas, mas também abordagens detalhadas para o processo de sistemas com RV [16] [17] [18] [22] [23] que visem critérios de usabilidade para esses sistemas distribuídos ao longo de sua concepção, projeto e implementação. Segundo Stanney [20], usabilidade em sistemas com RV está apenas começando a receber o foco de atenção necessário para identificação de uma taxonomia de critérios de usabilidade específicos para esses sistemas. Princípios de usabilidade tradicionais não consideram características específicas de sistema com RV. Além disso, princípios de usabilidade para sistema com RV dificilmente são encontrados e empiricamente derivados ou validados, bem como métodos adequados para esses sistemas são também dificilmente encontrados. Métodos de avaliação de usabilidade tradicionais têm sido aplicados em sistema com RV. No entanto, métodos de avaliação de usabilidade tradicionais podem não representar características específicas de sistema com RV, apresentando limitações quando aplicados nesses sistemas.
Nesse contexto, a MAUVE (Multi-criteria Assessment of Usability for Virtual Environments) [20], baseada sobre uma revisão e síntese de trabalhos mais recentes [5][10][11][24] para identificar diretivas de usabilidade em sistema com RV, apresenta uma hierarquia de critérios de usabilidade para esses sistemas [8], os quais são adotados por este estudo:

1. *Características de Usabilidade do Sistema com RV*
   - 1.1. Interação
     - 1.1.1. Navegação: localização e orientação prontamente.
     - 1.1.2. Manipulação de Objetos: seleção e manipulação de objetos naturalmente.
     - 1.1.3. Movimento do Usuário: controle de movimento intuitiva e continuamente.
   - 1.2. Entrada e Saída Multimodais de Sistema
     - 1.2.1. Visual: otimização do campo de visão e minimização de distorções.
     - 1.2.2. Auditivo: emprego e retorno auditivo eficaz, oportuna e significativamente.
     - 1.2.3. Tátil: emprego e retorno tátil e de força para apoiar tarefas confiavelmente.
2. *Considerações do Usuário do Sistema com RV*
   - 2.1. Envolvimento
     - 2.1.1. Presença: promoção de presença (experiência subjetiva de estar em um ambiente quando se está fisicamente em outro)
     - 2.1.2. Imersão: promoção de imersão (percepção de estar incluído em e interagindo com um ambiente que provê fluxo contínuo de experiências).
   - 2.2. Efeitos Colaterais
     - 2.2.1. Conforto: minimização de desconforto físico e maximização de segurança.
     - 2.2.2. Males: minimização de náusea, desorientação e distúrbios visuais.
     - 2.2.3. Efeitos Posteriores: minimização de efeitos posteriores.

Inserir esses critérios no processo de sistemas com RV se apresenta como desafio para a obtenção de sistema com RV que atenda amplamente as necessidades de seus usuários. Segundo Bevan [2], há uma necessidade de incorporar mais informações sobre qualidade de uso, usabilidade e projeto centrado no usuário em métodos para o processo de sistemas. Desse modo, devido a grande importância da usabilidade em sistemas com RV, sugere-se que critérios de usabilidade para sistemas com RV sejam incorporados ao processo desses sistemas.

3. Apresentação e Análise Crítica das Metodologias para o Processo de sistemas com RV Verificando as Implicações no Processo de sistemas com RV

Diversos contatos com os proponentes das quatro metodologias para sistemas com RV encontradas na literatura, VRID, CLEVR, de Stuart e de Scaife e Rogers, foram realizados através de e-mails para que a interpretação e a análise crítica realizada sob a ótica deste trabalho fosse validada sob a ótica dos autores dessas metodologias.

3.1 Metodologia VRID

Inicialmente, o modelo e metodologia VRID (Virtual Reality Interface Design) [22] contém direcionamentos para o projeto de interface de sistema com RV, sendo composta pelo modelo e pela metodologia VRID. Para segmentar e diferenciar a maior complexidade de objetos de interface de sistema com RV, o modelo VRID é organizado em torno de uma arquitetura de objetos multicomponentes. Para aplicar o modelo VRID no projeto de interfaces de sistema com RV, a metodologia VRID é formada por duas fases iterativas de projeto.

O modelo e metodologia VRID se apresenta como uma abordagem apenas para o projeto de interfaces de sistemas com RV e de forma restrita ao contexto da IHC, não considerando o enfoque no usuário ou em usabilidade em seu processo. A metodologia VRID associada ao modelo VRID, através de iterações e refinamentos entre suas fases de projeto, permite obter projetos de interface de sistema com RV conceitualmente seguros e implementáveis na prática. No entanto, a metodologia VRID não apresenta algumas atividades esperadas para abranger todo o processo de sistemas com RV (como análise de requisitos, implementação e teste), bem como outros componentes que consistem esses sistemas (como usuário e aplicação), os quais tornariam essa metodologia mais completa.

3.2 Metodologia CLEVR

A segunda abordagem, a metodologia CLEVR (Concurrent and Level by Level Development of VR Systems) [17], direcionada ao processo de sistemas com RV como um todo e baseada no paradigma Espiral da ES [14], é formada por três etapas nas quais análise de requisitos, projeto e validação são realizadas iterativamente.

A metodologia CLEVR se mostra como uma abordagem para todo o processo de sistemas com RV, porém também de forma restrita ao contexto da IHC, não incorporando o enfoque no usuário ou em usabilidade em seu processo proposto. O processo proposto nessa metodologia, conforme seus autores, está baseado no modelo de processo Espiral da ES. No entanto, percebe-se que não é possível identificar claramente como essa metodologia está fundamentada sobre esse modelo de processo, uma vez que não são apresentadas algumas das atividades características desse modelo de processo (como análise de riscos) durante todo o processo proposto. As atividades a serem realizadas, bem como o momento de sua realização, são bem definidos na metodologia CLEVR, possibilitando que a complexidade associada ao processo de sistemas com RV seja melhor gerenciada. Porém, devido ao desenvolvimento incremental e ao desenvolvimento hierárquico adotados nessa metodologia, segundo seus autores, ou baseado no modelo de processo Evolucionário da ES, conforme este estudo, suas etapas e iterações são realizadas de tal modo que não é possível determinar precisamente quais atividades pertencem a que etapa e ou iteração. Acredita-se que para que a metodologia CLEVR se torne mais completa, no âmbito das exigências no processo de sistemas com RV, é fundamental direcionar seu processo para o enfoque no usuário e em usabilidade.

Desse modo, com base na análise crítica deste estudo e nos diversos contatos com os autores da metodologia CLEVR, tem-se uma nova interpretação e representação para essa metodologia, mostrada na figura 1, sugerida por este trabalho e ratificada pelos autores dessa metodologia.

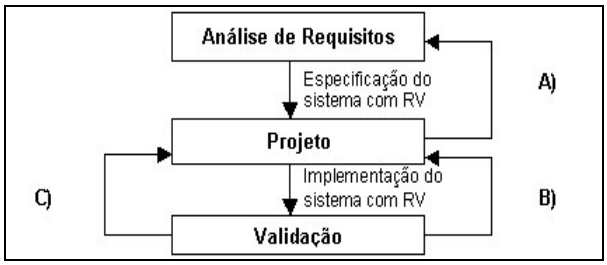


Figura 1 - Nova Interpretação e Representação para a Metodologia CLEVR

Pela figura 1, nessa nova interpretação e representação da metodologia CLEVR, suas iterações de requisito, projeto e validação são mantidas através da percepção de cada passo dessas iterações como uma etapa no processo dessa metodologia e três ciclos que permitem iteratividade são identificados e acrescentados. Esses ciclos eliminam a necessidade de se possuir um conjunto completo de especificações de requisitos para o sistema com RV antes do início de sua implementação, mantendo o que é chamado de desenvolvimento incremental pelos autores dessa metodologia,

bem como consideram hierarquicamente características críticas de forma, função e comportamento de objetos, mantendo o que é chamado de desenvolvimento hierárquico pelos autores dessa metodologia, e de maneira bottom-up no processo de sistemas com RV. Nessa nova interpretação e representação da metodologia CLEVR, os resultados encontrados ao final de cada etapa são acrescentados. Os ciclos são descritos a seguir:

A. Análise de Requisitos: aspectos usuais e da tecnologia de RV como identificação dos objetos e cenas, modelagem e refinamento da hierarquia de classes, das formas, funções e comportamentos dos objetos e das interações.

B. Projeto: simulação incremental para validar os modelos de especificação e predizer o desempenho do sistema e para transformar esses modelos de especificação em um modelo de implementação que possa ser compilado e executado na plataforma alvo.

C. Validação: melhoria da característica de imersão, decidindo empregar ou não elementos adicionais conhecidos por afetar a sensação de presença como aumento das entradas e saídas, variação das formas para o processo de interação, minimização da distração e utilização de efeitos especiais.

Essa nova proposta reproduz o processo de sistemas com RV sugerido pela metodologia CLEVR de modo fiel à sua aplicação, apesar de não sugerir o modelo de processo Espiral da ES como fundamentação, o qual, conforme a análise crítica realizada neste trabalho, não representa fielmente a fundamentação da metodologia CLEVR.

3.3 Metodologia de Stuart

A terceira abordagem estudada, a metodologia de Stuart [21], contém direcionamentos para a análise e para o projeto de sistemas com RV, sendo baseada na metodologia conhecida como Projeto Iterativo [7], é composta por quatro passos iterativos de projeto.

A metodologia de Stuart se apresenta como uma abordagem apenas para a análise e para o projeto de sistemas com RV e de modo mais amplo ao contexto da IHC, ou seja, incorporando o enfoque no usuário e em usabilidade. O processo proposto nessa metodologia, segundo esta pesquisa, está baseado na abordagem PCU da IHC, que por sua vez está baseado no modelo de processo Prototipação da ES, considerando, desse modo, o enfoque no usuário. A preocupação com usabilidade é considerada na metodologia de Stuart durante as avaliações de protótipos realizadas com usuários. Os passos que consistem essa metodologia são abrangentes, descrevendo características de sistemas com RV importantes somente em um nível mais baixo de abstração no processo de sistemas com RV. No entanto, a metodologia de Stuart não apresenta claramente alguns passos esperados para contemplar todo o processo de sistemas com RV (como implementação, avaliação e correção de possíveis erros) que consistem esses sistemas, os quais tornariam essa metodologia mais completa.

Dessa forma, com base na análise crítica deste trabalho e nos diversos contatos com o autor da metodologia de Stuart, tem-se uma nova interpretação e representação para essa metodologia, apresentada na figura 2, sugerida por este trabalho.

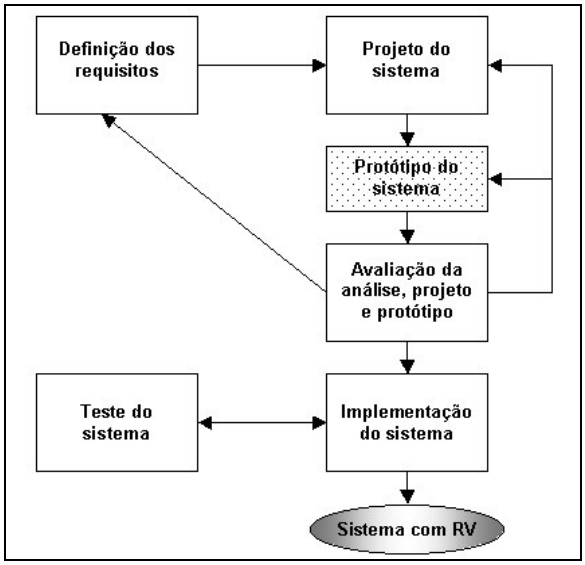


Figura 2 - Nova Interpretação e Representação para a Metodologia de Stuart

Conforme a figura 2, nessa nova interpretação e representação para a metodologia de Stuart, os passos de Definição de requisitos, Prototipação, Projeto e Avaliação são mantidos, dois outros passos e dois ciclos são acrescentados. As

etapas de Implementação do sistema e Teste do sistema permitem que as atividades de implementação, testes e correções de possíveis erros estejam claramente inseridas e representadas nessa metodologia, a qual, desse modo, abrange todo o processo de sistemas com RV. Os dois ciclos acrescentados a partir do passo de Avaliação em direção aos passos de Prototipação e de Projeto permitem que as possíveis iterações decorrentes da necessidade de se ajustar não apenas a análise de requisitos, mas ainda o projeto e ou o protótipo, também estejam claramente inseridas e representadas nessa metodologia. Nessa nova interpretação e representação da metodologia de Stuart, os resultados encontrados ao final de cada etapa são acrescentados.
Essa nova proposta reproduz o processo de sistemas com RV sugerido pela metodologia de Stuart de modo fiel à sua aplicação, apesar de inserir ciclos e passos, os quais, conforme a análise crítica deste trabalho, não estavam representados na metodologia de Stuart.

3.4 Metodologia de Stuart

Por fim, a abordagem de Scaife e Rogers [16], também direcionada ao processo de sistemas com RV como um todo e baseada na abordagem de PCU, é formada por cinco etapas nas quais atividades são realizadas progressiva ou paralelamente.
A metodologia de Scaife e Rogers se mostra como uma abordagem para todo o processo de sistemas com RV se comparada com a metodologia de Stuart, porém também de modo mais amplo ao contexto da IHC, ou seja, incorporando o enfoque em usabilidade e no usuário. O processo proposto nessa metodologia, conforme seus autores, está baseado na abordagem de PCU da IHC, que por sua vez está baseado no modelo de processo Prototipação da ES, considerando, desse modo, o enfoque no usuário. A preocupação com usabilidade é ponderada na metodologia de Scaife e Rogers durante as avaliações de protótipos realizadas com usuários. No entanto, ciclos que permitem iteratividade de projeto em seu processo não são encontrados claramente nessa metodologia. Acredita-se que para que a metodologia de Scaife e Rogers se torne mais completa, no âmbito das exigências no processo de sistemas com RV, é fundamental inserir iteratividade em seu processo.
Desse modo, com base na análise crítica deste estudo e nos diversos contatos com o autor da metodologia de Scaife e Rogers, tem-se uma nova interpretação e representação para essa metodologia, mostrada na figura 3, sugerida por este trabalho e ratificada pelos autores dessa metodologia.

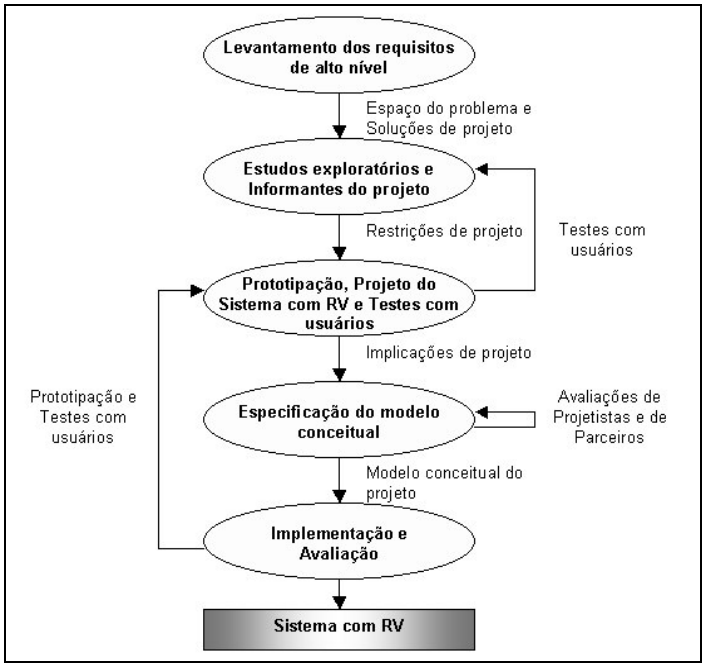


Figura 3 - Nova Interpretação e Representação para a Metodologia de Scaife e Rogers

Pela figura 3, nessa nova interpretação e representação da metodologia de Scaife e Rogers, suas cinco etapas são mantidas e três ciclos que permitem iteratividade são identificados e acrescentados. As atividades realizadas em cada etapa são mantidas na nova interpretação dessa metodologia, porém suprimidas na nova representação da metodologia de Scaife e Rogers, pois a realização dessas atividades é fundamental no processo proposto pela metodologia de Scaife e Rogers, mas sua representação não contribui para o entendimento claro e objetivo dessa metodologia. Nessa nova

interpretação e representação da metodologia de Scaife e Rogers, os resultados encontrados ao final de cada etapa são acrescentados. Os ciclos são descritos a seguir:

- **Testes com Usuários:** requisitos não especificados inicialmente podem ser descobertos nos testes com usuários, exigindo novos estudos exploratórios e novas entrevistas com informantes de projeto.
- **Avaliação de Projetistas e de Parceiros:** avaliações entre projetistas e parceiros de projeto são realizadas para que o modelo conceitual seja especificado.
- **Prototipação e Teste com Usuários:** novos protótipos podem ser construídos e utilizados nos testes com usuários, possibilitando que requisitos de projeto sejam novamente refinados.

Essa nova proposta reproduz o processo de sistemas com RV sugerido pela metodologia de Scaife e Rogers de modo fiel à sua aplicação, atendendo as necessidades de uma estrutura iterativa de processo, a qual, conforme a análise crítica deste trabalho, não estava presente na metodologia de Scaife e Rogers.

4. Análise comparativa das metodologias para sistemas com RV Verificando as Implicações no Processo de sistemas com RV

Como resultado da análise comparativa entre as metodologias para sistemas com RV citadas na literatura, é apresentada a tabela 1, que mostra sobre quais exigências do processo de sistemas com RV as metodologias para sistemas com RV estão fundamentadas.

Metodologia para Sistemas com RV	Enfoque no Usuário	Iteratividade de Projeto	Critérios de Usabilidade	Abrangência
VRID	Não	Sim	Não	Projeto de interfaces com RV
CLEVR	Não	Sim	Não	Todo o processo de sistemas com RV
Stuart	Sim	Sim	Não	Análise e projeto de sistemas com RV
Scaife e Rogers	Sim	Não	Não	Todo o processo de sistemas com RV

Tabela 1 - Tabela Comparativa das Implicações no Processo das Metodologias para Sistemas com RV

Conforme a tabela 1, apenas as metodologias de Scaife e Rogers e de Stuart consideram o enfoque no usuário e em usabilidade para o processo de sistemas com RV, realizando prototipações e avaliações com usuários. Porém, a metodologia de Stuart não explicita claramente atividades de implementação, avaliação e correção de possíveis erros que complementem o projeto gerado em seu processo, as quais são esperadas para completar o processo de sistemas com RV, ao passo que a metodologia de Scaife e Rogers não explicita claramente ciclos que gerem iteratividade em seu processo, a qual é esperada para o processo de sistemas com RV.

5. Meta-modelo para o Processo de sistemas com RV Considerando Enfoque no Usuário, Iteratividade de Projeto e Critérios de Usabilidade

A construção do meta-modelo para sistemas com RV é iniciada através da análise das contribuições de fases ou etapas ou passos, bem como atividades que compõem as metodologias para sistemas com RV encontradas na literatura.

O enfoque no usuário necessário para o meta-modelo é sugerido a partir de sua fundamentação sobre a abordagem do PCU ou do DP da IHC e, conseqüentemente, sobre o modelo de processo Prototipação da ES.

A iteratividade necessária para o meta-modelo é sugerida a partir da iteratividade encontrada através das fases, passos ou etapas que compõem as metodologias para sistema com RV, determinando os ciclos inseridos através das etapas do meta-modelo para sistemas com RV proposto, assim como as atividades necessárias para cada etapa do meta-modelo para sistemas com RV proposto são sugeridas a partir da análise comparativa das atividades que compõem cada fase, passo ou etapa das metodologias para sistema com RV, determinando as atividades inseridas através das etapas do meta-modelo para sistemas com RV proposto.

Desse modo, o meta-modelo para sistema com RV proposto por este trabalho é dividido em cinco etapas, representadas na figura 4.

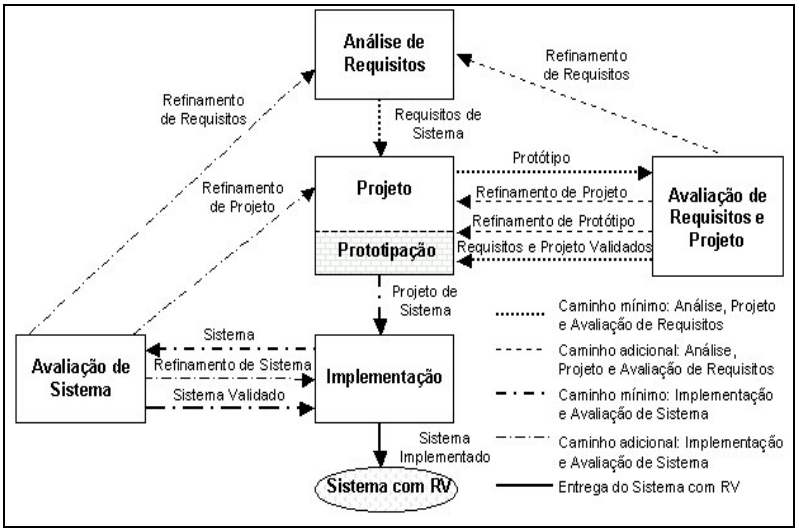


Figura 4 - Meta-modelo para Sistemas com RV com Enfoque no Usuário e Iteratividade de Projeto

Inicialmente, pela figura 4, são realizadas a etapa de Análise de Requisitos, a etapa de Projeto (com base em prototipação) e a etapa de Avaliação de Requisitos e Projeto.

Na etapa *Análise de Requisitos* devem ser realizadas as atividades: definir e delimitar o espaço do problema; identificar os benefícios da utilização da tecnologia de RV para o espaço do problema delimitado; identificar e analisar as pesquisas, teorias e possibilidades tecnológicas existentes para o espaço do problema; definir os usuários, suas capacidades, as tarefas e o ambiente; definir os requisitos funcionais, práticos e específicos do sistema com RV; solicitar aos informantes de projeto suas visões do espaço do problema delimitado; realizar estudos exploratórios, identificando as práticas atuais das formas de representação , dos mecanismos e dispositivos de interação e controle, da melhor combinação entre eles e as possibilidades futuras no espaço do problema; refinar os requisitos do sistema com RV conforme o resultado da etapa de Avaliação de Requisitos e Projeto.

A etapa *Projeto (baseado em Prototipação)* envolve as atividades, conforme a análise resultante da etapa de Análise de Requisitos: construir protótipos do sistema com RV, com ou sem o apoio de tecnologia computacional como em papel ou em ferramenta computacional para prototipação em RV, retratando funcionalidades e componentes de interface e de interação identificados; descartar idéias de projeto com base no resultado da etapa de Avaliação de Requisitos e Projeto; especificar o modelo conceitual do sistema com RV; discutir e refinar o modelo conceitual do sistema com RV com parceiros de projeto; projetar ou selecionar as tecnologias de entrada e saída, a arquitetura e as tecnologias computacionais; especificar ou modelar as funcionalidades do sistema com RV; especificar ou modelar os componentes de interface e de interação do sistema com RV; construir protótipos do sistema com RV, com apoio de tecnologia computacional como ferramenta computacional para prototipação em RV, retratando funcionalidades e componentes de interface e de interação projetados; refinar o projeto do sistema com RV conforme o resultado da etapa de Avaliação de Requisitos e Projeto.

Na etapa *Avaliação de Requisitos e Projeto* devem ser realizadas as atividades: testar os protótipos do sistema com RV utilizando usuários ou profissionais de avaliação; avaliar o desempenho do protótipo e do usuário; avaliar a usabilidade do protótipo; reportar os problemas de requisitos, projeto e ou do protótipo, determinando se há iteração a partir da etapa de Análise de Requisitos ou de Projeto (baseado em prototipação).

Essas três etapas iniciais formam o primeiro ciclo que permite iteratividade de projeto (composto pelos caminhos mínimo e adicional para Análise, Projeto e Avaliação de Requisitos na figura 4) do meta-modelo para sistemas com RV. Esse primeiro ciclo se encerra quando os requisitos do sistema com RV e o projeto são validados através da construção do protótipo e da sua avaliação de modo satisfatória por desenvolvedores e usuários do sistema com RV ou profissionais especializados em avaliação ou, ao menos por usuários do sistema com RV ou profissionais da IHC especializados em avaliação, garantindo o enfoque no usuário através do PCU ou o DP (seta nomeada Requisitos e Projeto Validados na figura 4). Nota-se que caso a avaliação do protótipo indique problemas na definição dos requisitos, as etapas de Análise de Requisitos, Projeto (com base em prototipação) e de Avaliação de Requisitos e Projeto devem ser realizadas novamente. Caso a avaliação do protótipo indique problemas na especificação de projeto, as etapas de Projeto (com base em prototipação) e de Avaliação de Requisitos e Projeto devem ser realizadas novamente. E, caso a avaliação do protótipo indique problemas na implementação do protótipo, caso não esteja de

acordo com os requisitos definidos ou com o projeto especificado ou caso apresente erros de execução, as etapas de Projeto (com base em prototipação) e de Avaliação de Requisitos e Projeto devem ser realizadas novamente.
Finalmente, ainda pela figura 4, são realizadas a etapa de Implementação e a etapa de Avaliação de Sistema.
A etapa *Implementação* envolve as atividades, conforme o projeto resultante da etapa de Projeto: implementar o projeto do sistema com RV; corrigir o sistema com RV conforme o resultado da etapa de Avaliação de Sistema.
Na etapa de *Avaliação de Sistema* devem ser realizadas as atividades: testar o sistema com RV utilizando usuários ou profissionais de avaliação; avaliar o desempenho do sistema; avaliar o desempenho do usuário; avaliar a usabilidade do sistema; reportar os erros do sistema e os problemas ainda remanescentes de requisitos, projeto e ou do protótipo, determinando se há iteração a partir da etapa de Análise de Requisitos, de Projeto (baseado em prototipação) ou de Implementação.
Essas duas etapas finais formam o segundo ciclo que permite iteratividade de projeto (composto pelos caminhos mínimo e adicional para Implementação e Avaliação de Sistema na figura 4) do meta-modelo para sistema com RV. Esse segundo ciclo se encerra quando o sistema é validado de forma satisfatória através da implementação do sistema e da sua avaliação de modo satisfatória por desenvolvedores e usuários do sistema com RV ou profissionais da IHC especializados em avaliação ou, ao menos por usuários do sistema com RV ou profissionais especializados em avaliação, garantindo o enfoque no usuário através do PCU ou o DP (seta nomeada Sistema Validado na figura 4). Nota-se que caso a avaliação do sistema ainda indique problemas na definição dos requisitos ou na especificação do projeto, o primeiro ciclo, a partir, respectivamente, da etapa Análise de Requisitos ou Projeto (com base em prototipação), deve ser realizado novamente e, posteriormente, as etapas de Implementação e Avaliação do Sistema também devem ser realizadas novamente. E, caso a avaliação do sistema indique problemas na implementação do sistema, o qual não está de acordo com os requisitos definidos ou com o projeto especificado ou apresenta erros de execução, as etapas Implementação e Avaliação do Sistema devem ser realizadas novamente.
A construção do meta-modelo para sistemas com RV é iniciada através da análise das possíveis contribuições de fases ou etapas ou passos, bem como atividades que compõem as metodologias para sistemas com RV encontradas na literatura, como representadas nas tabelas .
A construção do meta-modelo para sistemas com RV é finalizada através da análise dos critérios de usabilidade que devem ser considerados ou avaliados através de suas etapas, como representados na figura 5.

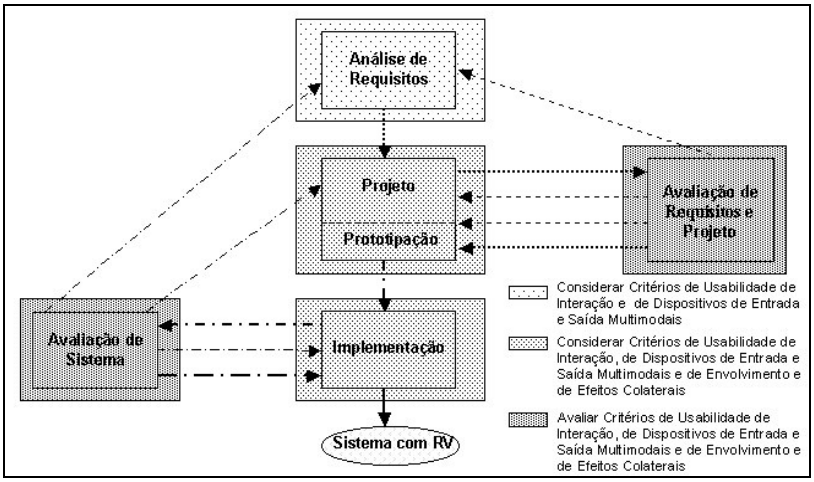


Figura 5 - Meta-modelo para Sistemas com RV com Enfoque no Usuário, Iteratividade de Projeto e Critérios de Usabilidade

Inicialmente, conforme a figura 5, de acordo com Stanney [20], o processo de interação deve ser natural, eficiente e apropriado para usuários, domínios e objetivos de tarefa do sistema com RV. Considerando-se que requisitos referentes ao processo de interação podem ser elicitados na etapa de Análise de Requisitos, acredita-se que se deve pensar em critérios de usabilidade de Interação na etapa de Análise de Requisitos do meta-modelo para sistemas com RV proposto. Uma importante característica de sistemas com RV é apresentar aos usuários múltiplas entradas e saídas através de dispositivos multimodais tais como visuais, auditivos e táteis [13] [20]. Todos esses modos de informação apresentados aos usuários devem ser prontamente entendidos, não ambíguos e necessários para atingir os objetivos das tarefas oferecidas pelo sistema com RV [20]. A escolha de exibição de parâmetros do sistema com RV através de dispositivos multimodais pode afetar significativamente a usabilidade de sistemas com RV [13]. Assumindo-se que requisitos referentes à exibição de parâmetros do sistema com RV e aos dispositivos multimodais podem ser

levantados na etapa de Análise de Requisitos, sugere-se que se deve considerar critérios de usabilidade de Dispositivos de Entrada e Saída Multimodais na etapa de Análise de Requisitos do meta-modelo.
O processo de interação e os dispositivos do sistema com RV são projetados na etapa de Projeto [5][17][21]. Considerando-se que assegurar que o processo de interação e os dispositivos atendam aos requisitos do sistema com RV é de fundamental importância para a usabilidade desse sistema, uma vez que permite aos usuários interagir com esse sistema sem que haja frustração e irritação [20], acredita-se que critérios de usabilidade de Interação e de Dispositivos de Entrada e Saída Multimodais devem ser pensados na etapa de Projeto (baseado em prototipação) do meta-modelo para sistema com RV. Outra importante característica de sistemas com RV é o envolvimento do usuário com esses sistemas, que é considerado de acordo com a sensação de imersão e de presença provida pelos sistemas com RV [15] [19]. Para proporcionar a sensação de imersão, critérios de usabilidade podem ser considerados no projeto de software e hardware de sistemas com RV, uma vez que essa sensação é diretamente afetada por pelo projeto de hardware e software desses sistemas [20]. Para facilitar a sensação de presença, critérios de usabilidade podem ser levantados considerando-se como melhor implementar os componentes de hardware e software do sistema com RV [15]. Assumindo-se que o envolvimento do usuário em sistemas com RV é influenciado pelo projeto de hardware e software desse sistema, e por como melhor implementar esses componentes do sistema com RV, sugere-se que critérios de usabilidade de Envolvimento devem ser considerados na etapa de Projeto (baseado em prototipação) do meta-modelo.
Finalmente, ainda conforme a figura 5, segundo Stanney [19], efeitos colaterais como desconforto físico ou distúrbios fisiológicos e psicológicos associados com o uso de sistemas com RV devem ser minimizados. Dentre esses efeitos, desconforto físico geralmente é associado com o ajuste e a aceitação de dispositivos utilizados pelos usuários no processo de interação com o sistema com RV. No entanto, como conforto e segurança do usuário no uso desses dispositivos são fatores críticos para a usabilidade de sistemas com RV [12], e como a escolha de dispositivos utilizados pelos usuários no processo de interação com o sistema com RV é realizada na etapa de Projeto [21], acredita-se que se deve considerar critérios de usabilidade de Efeitos Colaterais na etapa de Projeto (baseado em Prototipação) do meta-modelo para sistemas com RV.
Vale notar que nas etapas de Avaliação de Requisitos e Projeto, de Implementação e de Avaliação de Sistema do meta-modelo para sistemas com RV, todos os critérios de usabilidade propostos por Stanney [20] estão incorporados a esse meta-modelo, uma vez que todos já foram inseridos ao longo das etapas de Análise de Requisitos e Projeto e, desse modo, sugere-se que se deve pensar e avaliar todos os critérios de usabilidade, ou seja, de Interação, Dispositivos de Entrada e Saída Multimodais, Envolvimento e Efeitos Colaterais, nas etapas de Avaliação de Requisitos e Projeto, de Implementação e de Avaliação de Sistema.

6. CONCLUSÕES

A perspectiva da qualidade no uso é a visão da qualidade do sistema a partir do ponto de vista do usuário. O alcance a essa perspectiva pode ser considerado como um dos maiores objetivos no processo de sistemas interativo. Nesse contexto, sistemas com RV apresentam características mais específicas que podem afetar a qualidade no uso desses sistemas, aumentando a complexidade do processo de sistemas com RV e devendo ser consideradas sob essa perspectiva da qualidade ao longo da concepção, projeto e implementação desses sistemas.
No entanto, desenvolver sistema com RV que apresente valor estético, funcional e de uso, atendendo amplamente aos anseios de seus usuários, se apresenta como desafio, já que ainda precisam ser ponderadas inúmeras questões, dentre elas melhor analisar e formalizar o processo desses sistemas, considerando a perspectiva da qualidade no uso, a qual é focada por este estudo através de enfoque no usuário, de iteratividade de projeto e de critérios de usabilidade para sistemas com RV.
Dessa forma, um meta-modelo para o processo de sistemas com RV é proposto por este trabalho, fundamentado nas especificidades desses sistemas e nas implicações em seu processo através da consideração de princípios da IHC inseridos ao longo da concepção, projeto e implementação de sistemas com RV, a fim de garantir o estilo de interação que o emprego da tecnologia de RV em sistemas interativos objetiva prover.

References

[1] Bevan, N. Measuring usability as quality of use, in Journal of Software Quality 4, 115-130.

[2] Bevan, N. Quality in Use: Meeting User Needs for Quality, in Journal of Systems and Software (1999), 1-12.

[3] Bowman, D. Kruijff, E., La Viola, J., and Poupyrev, I. The Art and Science of 3D Interaction, in Tutorial notes from IEEE International Virtual Reality 2000 Conference (2000), 18-22.

[4] Brown, J. HCI and Requirements Engineering - Exploring Human-Computer Interaction and Software Engineering Methodologies for the Creation of Interactive Software, in ACM SIGHCI Bulletin 29, 1 (1997), 32-35.

[5] Gabbard, J.L., Hix, D., and Swan, E. User-centered design and evaluation of virtual environments, in IEEE Computer Graphics and Applications 19, 6 (1999), 51–59.

[6] Goransson B., and Sandback S. Usability designers improve the user-centred design process, in Proceedings for INTERACT'99 (1999), 1-4.

[7] Gould, J. D., and Levis, C. Designing for Usability: Key Principles and What Designers Think, in Communications of the ACM 28, 3 (1985), 300-311.

[8] User Guide Version 2.0. Multi-criteria Assessment of Usability for Virtual Environments, of Design Interactive Inc. (2004), 1-21.

[9] Jacob, R. J. K., Deligiannidis, L., and Morrison, S. A System Model and Specification Language for Non-WIMP User Interfaces, in ACM Transactions on Computer-Human Interaction 6, 1 (1999), 1-46.

[10] Kalawsky, R. S. VRUSE - A computerized diagnostic tool: for usability evaluation of virtual/synthetic environments systems, in Applied Ergonomics, 30 (1999), 11-25.

[11] Kennedy, R. S., and Lane, N. E., Berbaum, K. S., and Lilienthal, M. G. Simulator sickness questionnaire: An enhanced method for quantifying simulator sickness, in International Journal of Aviation Psychology 3, 3 (1993), 203-220.

[12] McCauley-Bell, P. Ergonomics in Virtual Environments. Handbook of Virtual Environments: Design, Implementation, and Applications (2002), 807-826.

[13] Mills, S., and Noyes, J. Virtual Reality: an overview of user-related design issues, in Interacting with computers 11, 4 (1999), 375-386.

[14] Pressman, R. System Engineering: A Practioner's Approach. New York: McGraw-Hill, 5 (2001).

[15] Sadowsky, W., and Stanney, K.M. Measuring and managing presence in virtual environments, in Handbook of Virtual Environments: Design, Implementation, and Applications (2002), 791-806.

[16] Scaife, M., and Rogers, Y. Informing the design of a virtual environment to support learning in children, in Journal of Human-Computer Studies 55, 2 (2001), 115-143.

[17] Seo, J., and Kim, G. J. Design for Presence: A Structured Approach to Virtual Reality System Design, in Presence: Teleoperators and Virtual Environments 11, 4 (2002), 378-403.

[18] Smith, S. P., and Harrison, M. D. Editorial: User centred design and implementation of virtual environments, in Journal of Human-Computer Studies 55, 2 (2001), 109-114.

[19] Stanney, K. M., Salvendy, G., Deisinger, J., Dizio, P., Ellis, S., and Ellison, J. Aftereffects and sense of presence in virtual environments: formulation of a research and development agenda, in International Journal of Human-Computer Interaction 10,2 (1998), 135-187.

[20] Stanney, K. M., Mollaghasemi, M., Reeves, L., Breaux, R., and Graeber, D. A. Usability engineering of virtual environments (VEs): identifying multiple criteria that drive effective VE system design, in International Journal of Human-Computer Studies 58, 4 (2003), 447-481.

[21] Stuart, R. Design of Virtual Environments. New York: McGraw-Hill, 1996.

[22] Tanriverdi, V., and Jacob, R. J. K. VRID: A Design Model and Methodology for Developing Virtual Reality Interfaces, in Proceedings of the ACM symposium on Virtual reality system and technology (2001), ACM Press, 175-182.

[23] Willians, J. S., and Harrison, M. D. A toolset supported approach for designing and testing virtual environment interaction techniques, in Journal of Human-Computer Studies 55, 2 (2001), 145-165.

[24] Witmer, B., and Singer, M. (1998). Measuring presence in virtual environment: a presence questionnaire, in Presence: Teleoperators and Virtual Environments 7, 3 (1993), 225-240.

Improvisational Multi-Agent Architecture: an Approach to Treat Unexpected Events Using Improvisation in Problem-Solving Process

Márcia Cristina Moraes
Universidade Federal do Rio Grande do Sul, Programa de Pós-Graduação em Computação
Pontifícia Universidade Católica do Rio Grande do Sul, Faculdade de Informática
Porto Alegre, Brazil, 90619-900
mmoraes@inf.ufrgs.br, mmoraes@inf.pucrs.br

and

Antônio Carlos da Rocha Costa
Universidade Federal do Rio Grande do Sul, Programa de Pós-Graduação em Computação
Pontifícia Universidade Católica de Pelotas, Escola de Informática
Pelotas, Brazil, 96010-000
rocha@atlas.ucpel.tche.br

Abstract

This paper aims to present an improvisational multi-agent architecture that includes improvisation as a capability for rational agents to handle problems that weren't anticipated in the planning of its course of action. Usually, when rational agents are in a problem-solving process they apply traditional re-planning techniques to solve unexpected problems. Re-planning techniques have several limitations such as possible exponential complexity and inadequacy for a world characterized by unpredictable events. Our proposal allows agents to give rapid answers to unexpected situations, independently of having explicit knowledge directly applicable to such situations. We allow agents to be capable of improvising behaviors using the resources readily available to them through an improvisation process based on analogy by similarity.

Keywords: Artificial Intelligence, Autonomous Agents and Multi-Agent Systems, Planning and Scheduling, Agent Architectures, Improvisational Agents

1 Introduction

Often, an artificial agent has the capabilities of reasoning, learning and communicating with other agents. Reasoning is used by a rational agent to solve problems in a complete and correct way. To do that rational agents plan their course of action in advance, trying to anticipate the future. However, real environments may change while an agent is reasoning about how to achieve some goal, and these changes may undermine the assumptions upon which the agent's reasoning is based. Agents in real, dynamic environments need to be receptive to many possible unexpected situations, which do not typically arise in a neatly sequential fashion. Agents need to reason about their actions [1]. To do that they have to know when new facts and opportunities happened and they have to adapt their selves to each current situation. Two alternatives broadly used in treating dynamic environments are probabilistic reasoning and re-planning that can be adapted through learning processes [2]. An alternative way, proposed by Hayes-Roth and Doyle [3] pointed out that when people are in front of unexpected situations, that demand a rapid and spontaneous answer, they use their capability for improvisation. For Hayes-Roth and Doyle in this kind of situation it is sometimes better to improvise than to re-plan.

The capability to produce spontaneous answers is particularly important for interface agents that can be personified through animated characters or human faces. Some researches have shown that users apply social rules to computers [4] [5]. So, as human agents, rational agents have to present coherent, interesting and believable behaviors, bringing the illusion of life to agents and making the user to suspend his disbelieves [6]. To do that, the agents have to be believable both in an expected and an unexpected situation.

To be believable in the first case, the agents can use what we call implicit improvisation. By implicit improvisation we mean the ability to incorporate predictable alternative courses of actions. In this way, we change planning to implicit improvisation because it is more reasonable to guide agents with an abstract course of behavior than to direct them with a complete course of actions. Implicit improvisation is treated using an approach based on constraint satisfaction.

In the other case, the agents can use what we call explicit improvisation. By explicit improvisation we mean the ability to treat an unpredictable known event using improvisation techniques explicitly. With explicit improvisation we aim to introduce the concept of improvisation in the problem-solving process of rational agents, treating improvisation in this case as a process based on analogy by similarity.

In this paper we present an improvisational multi-agent architecture that incorporates the changing of planning to implicit improvisation and the introduction of explicit improvisation in problem-solving for rational agents. The architecture is based on the ideas of Improvisational Theatre. So, agents can assume one of the three possible roles: director, actor and director-actor. The director-actor is a mixed role of the director and the actor. These roles imply two possible architecture's configurations: centralized and decentralized. The first one is composed by one director and several actors. The second one is composed by several directors-actors. In order to show how to incorporate the two kinds of improvisation in rational agents we are going to focus on the director's improvisation. As the process of implicit improvisation was already described in [7], this paper presents the explicit improvisation emphasizing the director's functions.

Related works on improvisation, such as Virtual Theatre Project [8], Oz Project [9] and Improv Project [10] have focused just on predictable known events, using approaches like planning and implicit improvisation. All these works focused the actors role and didn't exam the director role. Our proposal is a novelty because it treats explicit improvisation based on the directors' abilities of handling problems that weren't anticipated in the planning of the play.

2 Improvisation and Analogy by Similarity

According to Spolin [11], improvisation is related to the spontaneity to act in a world that is in constant motion. In this way, the idea of improvisation is implicitly appeals to informal and spontaneous behaviors, that had had no previous preparation. One of the first kinds of improvisational theatre was in the Commedia Dell'Arte. In Commedia, dramatic text was replaced by *canevas*, a kind of plan which contains only the main facts in a sequence that makes possible free improvisation by comedians [12]. In traditional theatre, on the other side, representation has always been thought of as something organized in advance. Chacra [13], however, pointed out that improvised and planned representation are only different poles of the same subject, determined by degrees that make the theatrical presentation more or less formalized or improvised. If actors intend to use improvisation they explicitly are integrated in what is called Improvisational Theater. So, they don't prepare in advance all their actions and speeches, they consider the moment of spontaneity. The moment of spontaneity acts in two kinds of improvisation already mentioned: implicit and explicit improvisation. Implicit improvisation involves text improvisation (which occurs through the set of phrases and non-verbal behaviors that are left open in the play for the free interpretation of the actors) and personality improvisation (which is related to the way an actor interprets one character, considering its various physical and psychological characteristics). Explicit improvisation involves problem-solving improvisation which occurs when an agent doesn't have a plan that can be immediately applied to an unexpected situation. In this

case, the director is responsible for helping actors to find solutions to unexpected problems. The director needs rapid answers to unexpected situations and he uses readily available resources to produce that answers. One approach that gives rapid answers and uses past experiences (resources) to produce new solutions for unexpected problems [14] is analogy by similarity. So, in order to handle problems that weren't anticipated, the agents implement problem solving improvisation as a process based on analogy by similarity.

In the absence of relevant specific information, traditional position in analogy studies has been that the most similar analogous brings, with higher probability, a correct solution. Russell [15] shows how a statistic analysis can be executed in order to produce a successful analogy, using only the supposition that there are some relevant characteristics in both source and target descriptions. Russell [15] and Davies [16] uses the theory of determination to provide a notion of relevance. This theory supports that as known similarities are (partially) relevant to inferred similarities, the analogical inference is guaranteed to be (partially) justified. In this way, Russell [14] proposes that at least one aspect of analogical reasoning consists of a probabilistic process of partial determinations. Through this probability the most similar analogous is guaranteed to be the most suitable analogy. When there is a higher number of relevant attributes it is necessary a closer overall match to ensure that relevant similarities are indeed present.

A simplified model for analogy in a database is: there is a target T described by m attribute-value pairs, for which we want to find a value for another attribute Q. There are several sources $S_1, ..., S_n$ (analogous) that have values for the attribute Q as well as for the m attributes known to for the target [14]. The similarity s is defined as the number of matching attribute values for a given target and source and the difference is defined by $d = m - s$. Assuming that there are r relevant attributes to find out the value of Q, $p(d,r)$ is defined as the probability that a source S, differing from the target in d attributes, matches it on the r relevant attributes. $p(d,r)$ is calculated using a simple combinatory argument [14]. Let N_m be the number of choices of which attributes are relevant such that S matches T on those attributes. Let N to be the total number of choices of relevant attributes. Then, [14] states that:

$$p(d, r) = N_m / N = \begin{bmatrix} m-d \\ r \end{bmatrix} \Big/ \begin{bmatrix} m \\ r \end{bmatrix} \quad (r >= 1)$$

This probability will guarantee that the most similar analogous is the most suitable analogy.

3 Improvisational Multi-Agent Architecture

This section presents an improvisational multi-agent architecture based on the ideas of Improvisational Theatre. In the proposed architecture, agents are responsible for the creation and presentation of some contents. In order to do that, they can assume one of the three possible roles: director, actor and director-actor. The director has to coordinate actors, informing to them their courses of actions, called scripts, and help actors to solve unexpected problems. The actors have to follow the director's instructions while improvising behaviors appropriate to each situation. The director-actor is a mixed role that agents can take, where they manage both responsibilities, of actors and directors. The architecture is improvisational in the sense that it includes improvisation in both actor and director functions. This makes possible the incorporation of implicit improvisation (including text and personality improvisation) related to predictable and known events and explicit improvisation (meaning problem-solving improvisation) related to unpredictable known events. Actors and director execute these kinds of improvisation in different levels of abstraction. The director is involved in the preparatory phase of improvisation and actors are involved in the execution phase.

Each agent, independently of its role, is organized around a two-level architecture. The higher-level contains processes related to the agents' cognitive capabilities and the lower-level, processes related to perception and action on the environment. In order to support the two kinds of improvisation mentioned before, the definition of the higher-level is based on the improvisational director's processes [11] and is composed by knowledge acquisition, intentions building and problem-solving improvisation, as shown in figure 1.

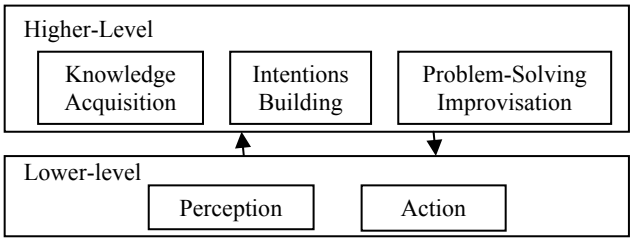


Figure 1: Improvisational Multi-Agent Architecture

Implicit improvisation is executed through knowledge acquisition and intentions building modules as a process of constraint satisfaction that already were described in [7]. The knowledge acquisition obtains information about the actors' courses of actions, composed by a sequence of, more or less abstract, activities and contents to be presented

through those activities. Based on that, the intentions building module constructs intentions for the actors. The intentions are also called abstract scripts because they are abstract plans that guide, but don't determine, the agent's behavior. Explicit improvisation is executed by problem-solving improvisation module as a process of analogy by similarity. The next sections present how explicit improvisation is performed in the problem solving process, considering the director role.

3.1 Problem-Solving Improvisation Module

After acquiring knowledge, building abstract scripts of behavior and sending them to actors, the director waits for a perception. A perception structure is composed by: a type of perception, agent identification (id) and object description. If the type of perception is fail, the agent id indicates which agent asked for problem-solving and which intention (abstract script of behavior) needs improvisation. The object description has the same set of attributes of a means object (see below). A fail action is generated by an actor when occurs an event that is unpredictable for its course of action. This event can generate two different situations. In the first case, it may happen that the agent has a piece of knowledge that can be directly applied to solve the problems cause by the situation. So it could use re-planning, although this it sometimes not the most appropriated solution, as we mentioned before. In the other case, re-planning is not applicable because the agent has no knowledge on how to behave in the situation. In both situations the problem can be solved by improvisation based on analogy by similarity.

3.1.1Ends Objects and Means Objects

We assume that, in an analogy, both source and target objects are described through a generic object model. In this model, the objects are of two kinds: ends and means. Each kind of object is described using sets of attributes.
Objects classified as ends are related to the agent's goal, in a specific course of action. Objects classified as means are related to the way a specific course of action is performed. The set of attributes for ends are: *objective*, *effect* and *particular characteristics*; and the set of attributes for means are: *possible uses* and *particular characteristics*. The possible uses group has a list of uses for an object. The particular characteristics group has a set of attribute-values pairs that specify the characteristics of the object. The values for attributes (objective, effect, list of possible uses and particular characteristics) depend on the application domain.
One ends object can be related to a means object by its particular characteristics. For instance, one ends object can have as particular characteristic the name of a means object that should be part of its course of action.

3.1.2 Kinds of Improvisation for Problem Solving

Improvisation for problem solving can occur in two ways: using an *improvised means* as an alternative way to reach the current goal or using an *improvised ends* as an alternative goal that reaches the current goal through some side-effects. In the first case we have improvisation at the level of means, and in the second case we have improvisation at the level of ends.

3.2 Problem-Solving Improvisation Module's Architecture

Figure 2 shows the architecture of the *problem-solving improvisation* module, with its inputs and output. The module receives as input the description of the object that provokes the fail, the objects that represent agent's knowledge and the intention that must be reconsidered. The description of a problem object follows the same pattern of a *means* type of object, so it is composed by possible uses and particular characteristics. Based on these inputs, the module executes the intention reconsideration using a process of analogy by similarity. To do that, the module architecture is composed by three sub-modules: *identification of improvisation kind*, *analogy building* and *transforming analogy into intention*.

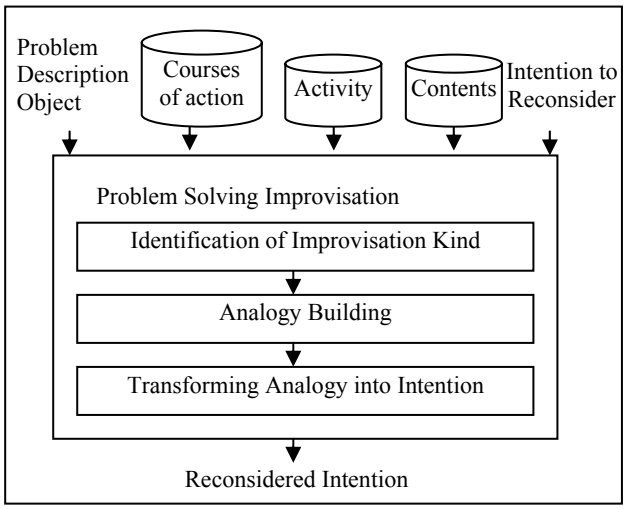


Figure 2: Problem-Solving Improvisation Module's Architecture

3.2.1 Identification of Improvisation Kind Sub-module

This sub-module receives the object representing the problem description and sends to the analogy building sub-module the type of improvisation to be executed, a list of possible analogous sources and the *problem description object*. In order to identify the kind of improvisation to be executed, the sub-module searches for *ends* objects (in the course of action objects databases) that have a goal compatible with one of the possible uses present in the *problem description object*. If one or more compatible objects are found, the kind of improvisation is of the *means* kind. In this moment, the sub-module searches for *means* objects (in the content objects database) that has possible uses and particular characteristics compatible with the *problem description object*. This process results in a list of possible analogous objects.

If it is no ends object are found, the kind of improvisation is ends improvisation. In this case, the sub-module searches for *ends* objects (in the course of action objects database), that has *effect* compatible with *possible uses* of the *problem description object*, building a list of possible analogous objects. If the list is empty, that is, there isn't any source object with effect compatible with *possible use* of *problem description* then we will have so-called improvisation without known effects. In this case, the sub-module searches for means objects (in the content objects database) that have *particular characteristics* similar to the problem object's *particular characteristics*. This process builds a list of possible analogous object.

In this way, the sub-module implements *means* improvisation and two kinds of *ends* improvisation: with known effect and without known effect. Both *means* and *ends with known effect* improvisations are due to a "strong analogy", while *ends without known effect* improvisation is due to a "weak analogy". Analogy is called strong when there is a relationship between possible uses of problem description objects and effects of source *ends* objects. Analogy is called weak when there is no such relationship, but there is some analogy between the particular characteristics. These two kinds of analogies always allow the agent to produce improvisations, even when the found analogy is not the most suitable one.

3.2.2 Analogy Building Sub-module

This sub-module uses the ideas presented by Russell [14][15] to select the most suitable analogous source object, given a target object description. So, we have to define the similarity *s*, the difference *d*, the relevant attributes *r* and the probability *p(d,r)* for each one of the possible analogous sources given by the identification of improvisation kind module.

The similarity *s*, target attributes *m* and relevant attributes *r*, are calculated for both *means* and *ends without known effect* improvisation, considering the set of attributes for particular characteristics and possible uses. In the *ends with known effect* improvisation, only the set of attributes of the *effect* kind is considered. This happens because when the goal changes, the particular characteristics need not necessarily be compatible, but the possible uses are compatible. In *means* improvisation, the current goal doesn't change, but the *means* can be different from the current one, so the sub-module takes the *possible uses* to search for similarities inside *means* objects.

After calculating the similarity of the analogous source, it must be calculated the difference among the *m* attributes that describe the target and a similarity *s*, as $d = m - s$. It is assumed *r* relevant attributes as an input argument and it is calculated *p(d,r)* as the probability of a source *S*, differing from its target in *d* attributes, matching with the *r* relevant attributes, using the formula presented in section 2. The supposition that there isn't any relevant information means that all attributes are equally relevant. The probability will guarantee that the most similar analogous is the

most suitable analogy. Although Russell proposed a probability function considering that *r* follows a probability distribution, we consider here that there is a fixed number of relevant characteristics *r*.
The sub-module applies the probability formula ($p(d,r) = N_m / N$) to each one of the possible analogous object contained in the set built by the *identification of the improvisation kind* sub-module. The chosen analogous object will be the one that has the highest probability of being similar to the target. At the end of its execution, the sub-module sends to the sub-module that transforms analogy into intention the kind of improvisation to be realized and the analogous source selected as the object most similar to the problem object.

3.3.3 Transforming Analogy into Intention Sub-module

This sub-module receives the kind of improvisation and the source object that is the most appropriate analogy to the problem description object and transforms this analogy into an intention. If the kind of improvisation is means or ends without known effect, the intention to be developed is an update of the current intention. This update will contain the source object most similar to the object that caused the problem. This similarity is calculated through the probability formula presented in section 2.
If the kind of improvisation is *ends with known effect*, the sub-module builds a new intention based on the *ends* object chosen. As mentioned before, an *ends* object has a relationship with *means* objects by its particular characteristics. The sub-module activates the intentions building module that starting from the *ends* object and its relations to *means* objects, builds the new intention and sends it back to the transforming analogy into intention sub-module.

4 Case Study: SAGRES Virtual Museum

The SAGRES system is a virtual museum that aims to support learning processes through the interaction among visitors and also between a visitor and the museum resources [17]. SAGRES facilitates the organization of visits to museums, presenting museum's information bases in a way adapted to the visitor's characteristics. The improvisational multi-agent architecture is being used to create virtual guides and personal assistants to SAGRES. At this moment, we have already done a simulation of the *problem-solving improvisation* module for the presentation of three different contents that are related to *means* and *ends* (*with and without know effect*). The first simulation involves the presentation of the content named *archeological sites*. So the description of the problem object that arrives for *problem-solving improvisation* module has the structure presented in table 1.

Object Description					
Possible Uses	Particular Characteristics				
	Kind	Area	Sub-area	Location	Floor
Archeological Sites Dinosaurs Archeology	not informed	History	In the Past	Exposition	not informed

Table 1: Description of the problem object named *archeological sites*

The *improvisation for identification kind* sub-module executes, as described in section 3.2.1, and looks, in the course of action database, for an object compatible with one of the possible uses described in the problem object. It finds a course of action, which objective is dinosaurs, that is compatible with the possible uses dinosaurs. In this way, the kind of improvisation that is going to be execute is *means* improvisation. The sub-module starts to build a list of possible analogous, searching in the content databases for objects that have particular characteristics and possible uses compatible with the problem object particular characteristics and possible uses. It finds the possible analogous objects described in table 2 to 5. The particular characteristics of a content represents information that the SAGRES system uses to present some content.

Possible Uses	Particular Characteristics					
	Kind	Area	Sub-area	Location	Floor	Contents
Before the dinosaurs Pre-history Dinosaurs Earth evolution	Specific	History	Million of years	Exposition	second	Before... The dinosaurs... Lived...

Table 2: Structure of the content named *before the dinosaurs*

Possible Uses	Particular Characteristics					
	Kind	Area	Sub-area	Location	Floor	Contents
In the past Dinosaurs Earth evolution	Specific	History	Million of years	Exposition	second	In the past... Million of years ago...

Table 3: Structure of the content *in the past*

Possible Uses	Particular Characteristics					
	Kind	Area	Sub-area	Location	Floor	Contents
In the present Dinosaurs Earth evolution	Specific	History	Million of years	Exposition	second	In the present... Nowadays...

Table 4: Structure of the content *in the present*

Possible Uses	Particular Characteristics					
	Kind	Area	Sub-area	Location	Floor	Contents
Archeological sites	Specific	History	In the Past	Exposition	second	The sites... Archeology... There are...

Table 5: Structure of the content *archeological sites*

The sub-module *improvisation for identification kind* sends the list of analogous objects (described in table 2 to 5) to the *analogy building* sub-module. This sub-module verifies the kind of improvisation and starts to calculate the similarity, difference and probability for each one of the possible analogous objects. As the improvisation kind is of *means*, the *analogy building* sub-module starts to calculate the similarity based on possible uses and particular characteristics. The similarity for objects presented in table 2, 3 and 4 is 3 because only the area and location characteristics are similar to the description of the problem object and the dinosaur possible use is present in all analogous objects. The difference for these objects is 2 because there are 5 attributes to describe the problem object. Considering that the number of relevant attributes is 2 the probability calculated by the formula $p(d,r) = N_m / N$ is:

$$= \begin{bmatrix} m-d \\ r \end{bmatrix} \Big/ \begin{bmatrix} m \\ r \end{bmatrix} = \begin{bmatrix} 5-2 \\ 2 \end{bmatrix} \Big/ \begin{bmatrix} 5 \\ 2 \end{bmatrix} = \frac{3!}{2!(3-2!)} \Big/ \frac{5!}{2!(5-2)!} = 0{,}3$$

The similarity for the object described in table 5 is 4 and the difference is 1. Considering that the number of relevant attributes is 2 the probability calculated by the formula $p(d,r) = N_m / N$ is:

$$= \begin{bmatrix} m-d \\ r \end{bmatrix} \Big/ \begin{bmatrix} m \\ r \end{bmatrix} = \begin{bmatrix} 5-1 \\ 2 \end{bmatrix} \Big/ \begin{bmatrix} 5 \\ 2 \end{bmatrix} = \frac{4!}{2!(4-2!)} \Big/ \frac{5!}{2!(5-2)!} = 0{,}6$$

After examining all the object present in the list of possible analogous object, the sub-module *analogy building* chooses the object that has the greater similarity. In this case, the chosen object was the object described in table 5. The chosen object and the kind of improvisation are send to the next sub-module, *transforming analogy into intention*. As the kind of improvisation is of *means* the current intention has to be updated. *Transforming analogy into intention* activates the *intentions building module* to do the intention's update.

The second simulation involved the presentation of the content named *meteors*. So the description of the problem object that arrives for *problem-solving improvisation* module has the structure presented in table 6.

Object Description					
Possible Uses	Particular Characteristics				
	Kind	Area	Sub-area	Location	Floor
Meteors Planets	not informed	History	Earth Planet	Exposition	Not informed

Table 6: Description of the problem object named *meteors*

The *improvisation for identification kind* sub-module executes and find out that there isn't any objective in the course of action database that is compatible with one of the possible uses in the description of the problem object. In this way, the sub-module searches for objects in the course of action database that have some effect compatible with one of the possible uses in the description of the problem object. As there isn't any, the kind of improvisation is of *ends without known effect*. So the sub-module searches in the content database for some object that has possible uses and particular characteristics compatible with the particular characteristics of the problem object description. The resultant list is composed by objects described in table 2, 3, 4, 5 and 7.

Possible Uses	Particular Characteristics					
	Kind	Area	Sub-area	Location	Floor	Contents
Minerals	Specific	History	Earth Planet	Exposition	second	The minerals... Many year ago... Many minerals...

Table 7: Structure of the content *minerals*

The sub-module *improvisation for identification kind* sends the list of analogous objects (described in table 2, 3, 4, 5 and 7) to the *analogy building* sub-module. This sub-module verifies the kind of improvisation and starts to calculate the similarity, difference and probability for each one of the possible analogous objects. As the improvisation kind is of *ends without known effect,* the *analogy building* sub-module starts to calculate the similarity based on possible uses and particular characteristics. The similarity for objects presented in table 2, 3, 4 and 5 is 3 because only the area, location and floor characteristics are similar to the problem object description. The difference for these objects is 2 because there are 5 attributes to describe the problem object. Considering that the number of relevant attributes is 2 the probability calculated by the formula $p(d,r) = N_m / N$ is:

$$= \begin{bmatrix} m-d \\ r \end{bmatrix} \Big/ \begin{bmatrix} m \\ r \end{bmatrix} = \begin{bmatrix} 5-2 \\ 2 \end{bmatrix} \Big/ \begin{bmatrix} 5 \\ 2 \end{bmatrix} = \frac{3!}{2!(3-2!)} \Big/ \frac{5!}{2!(5-2)!} = 0{,}3$$

The similarity for the object described in table 7 is 4 and the difference is 1. Considering that the number of relevant attributes is 2 the probability calculated by the formula $p(d,r) = N_m / N$ is:

$$= \begin{bmatrix} m-d \\ r \end{bmatrix} \Big/ \begin{bmatrix} m \\ r \end{bmatrix} = \begin{bmatrix} 5-1 \\ 2 \end{bmatrix} \Big/ \begin{bmatrix} 5 \\ 2 \end{bmatrix} = \frac{4!}{2!(4-2!)} \Big/ \frac{5!}{2!(5-2)!} = 0{,}6$$

After examining all the object present in the list of possible analogous object, the sub-module *analogy building* chooses the object that has the greater similarity. In this case, the chosen object was the object described in table 7. The chosen object and the kind of improvisation are send to the next sub-module, *transforming analogy into intention*. As the kind of improvisation is of *ends without known effect* the current intention has to be updated. *Transforming analogy into intention* activates the *intentions building module* to do the intention's update.

The third simulation involved the presentation of the content named *human and primate evolution*. So the description of the problem object that arrives for *problem-solving improvisation* module has the structure presented in table 8.

Object Description					
Possible Uses	Particular Characteristics				
	Kind	Area	Sub-area	Location	Floor
Human and primate evolution Earth evolution Pre-history	not informed	History	In the past	Exposition	Not informed

Table 8: Structure of the content *human and primate evolution*

The *improvisation for identification kind* sub-module executes and find out that there isn't any objective in the course of action database that is compatible with one of the possible uses in the description of the problem object. In this way, the sub-module searches for objects in the course of action database that have some effect compatible with one of the possible uses in the description of the problem object. The sub-module finds two compatible objects, described in table 9 and 10. This kind of improvisation is *ends with known effect*.

Objective	Effect	Initial Activity
Dinosaurs	Pre-history	Virtual guide presentation

Table 9: Structure of the course of action *earth stages of evolution*

Objective	Effect	Initial Activity
Earth Stages of Evolution	Earth evolution Human and primate evolution	Archeological site

Table 10: Structure of the course of action *earth stages of evolution*

The sub-module *improvisation for identification kind* sends the list of analogous objects (described in table 9 and 10) to the *analogy building* sub-module. This sub-module verifies the kind of improvisation and starts to calculate the similarity, difference and probability for each one of the possible analogous objects. As the improvisation kind is of *ends without effect knwon,* the *analogy building* sub-module starts to calculate the similarity based on possible uses of the problem object description, comparing it with the effect of the possible analogous objects. The object presented in table 9 has similarity equal to 1, because only one of the possible uses of the problem object description is in the list of this object. As *m* is equal to 3, because there are three possible uses in the problem object description, the difference is 2. Considering that the number of relevant attributes is 1 the probability calculated by the formula $p(d,r) = N_m / N$ is:

$$= \begin{bmatrix} m-d \\ r \end{bmatrix} \Big/ \begin{bmatrix} m \\ r \end{bmatrix} = \begin{bmatrix} 3-2 \\ 1 \end{bmatrix} \Big/ \begin{bmatrix} 3 \\ 1 \end{bmatrix} = \frac{1!}{1!(0!)} \Big/ \frac{3!}{1!(3-1)!} = 0{,}33$$

The similarity for the object described in table 10 is 2 because two of the possible uses of the problem object description are in the list of this object. The difference is 1. Considering that the number of relevant attributes is 1 the probability calculated by the formula $p(d,r) = N_m / N$ is:

$$= \begin{bmatrix} m-d \\ r \end{bmatrix} \Big/ \begin{bmatrix} m \\ r \end{bmatrix} = \begin{bmatrix} 3-1 \\ 1 \end{bmatrix} \Big/ \begin{bmatrix} 3 \\ 1 \end{bmatrix} = \frac{2!}{1!(2-1!)} \Big/ \frac{3!}{1!(3-1)!} = 0{,}66$$

After examining all the object present in the list of possible analogous object, the sub-module *analogy building* chooses the object that has the greater similarity. In this case, the chosen object was the object described in table 10. The chosen object and the kind of improvisation are send to the next sub-module, *transforming analogy into intention*. As the kind of improvisation is of *ends with known effect* the current intention has to be removed and a new one has to be created. *Transforming analogy into intention* activates the *intentions building module* to build the new intention.

Analyzing the *means* and *ends* (*with and without known effect*) kinds of improvisation implemented in the *problem-solving improvisation* module we conclude that the *problem-solving improvisation* produces coherent answers for the tests applied, always choosing the source object that correspond to the most similar analogous.

As previously considered in section 3.2.1, improvisation of *ends without known effect* can produce a "weak analogy" and it does happened in our simulation. Although the produced answer was not directly related to the asked subject, it did have some relationship between the particular characteristics of the subject, indicating that they are in the same super set of contents. This shows that even when agents doesn't have complete knowledge of the subject, they can produce some plausible answer preventing their fail of execution.
In the "weak analogy" case, agents can express their lack of specific knowledge explaining that they don't know that subject but they do know another one that is similar to what was requested in certain characteristics. Doing that, agents can act like humans in the same situation, showing smart and believable behavior and consequently producing the so desired illusion of life.

5 Final Considerations

This paper presented one approach to include improvisation concept in problem-solving process for rational agents through the specification of an improvisational multi-agent architecture. Improvisation is treated like a process of analogy by similarity due to the strong relationship between these two concepts. We consider analogy and case-based reasoning as synonymous and analogy by similarity was chosen because we believe that its the theory of determination, that provides a notion of relevance, is a suitable methodology for improvisation.

The proposed architecture is based on a traditional BDI architecture [18] and can be mapped into a BDI diagrammatic architecture as presented in Wooldridge [19] with one extension. The knowledge acquisition process represents the beliefs revision function. The intentions building contains both options and filter function. Perception represents input sensor and action represents action function. The extension is present in problem-solving improvisation that contains the characteristics of both options and filter functions to build an alternative course of action through analogy by similarity when something unexpected happens by accident and the agent doesn't know how to solve it.

Showing the mapping between our proposed architecture and Wooldridge's diagrammatic BDI architecture is important in order to insert improvisation as a fundamental characteristic of rational agents, improving their capabilities to handle problems that weren't anticipated in the planning of its course of action. The choose of a BDI architecture to show the possibility of using improvisation in rational agent has made due to the significance and recognition of BDI to describe rational agents behaviors. This overcomes the existent gap between improvisation and artificial intelligence and shows that improvisation is a natural problem-solving technique.

With the *problem-solving improvisation* module we extend the works previously done on improvisation, proposing an approach to treat unexpected know events through a director agent. Previous works only have concerned actors agents that are able to handle expected know situations trough planning and implicit improvisation.

The simulation performed in the SAGRES virtual museum shows that the *problem solving improvisation* module can produce relevant answers to unpredictable known events. However, that was only a simulation and it is not possible to do considerations about the efficiency of the *problem-solving improvisation* module. We intend to do other tests in order to compare the performances of the virtual guides' architecture with and without the improvisation modules and observe more complete and reliable results, including observations on the agent's ability to solve unexpected problems.

References

[1] Pollack, M. E. The use of plans. Artificial Intelligence, Vol. 57. Elsevier Science Publishers (1992) 43-68

[2] Russell, S.; Norvig, P. Artificial Intelligence: a modern approach. Upper Saddle River. (2003)

[3] Hayes-Roth, B., Doyle, P. Animated Characters. In: Autonomous Agents and Multi-Agent Systems, Vol. 1, Kluwer Academic Publishers, (1998) 195-230

[4] Ball, G.; Ling, D.; Kurlander, D.; Miller, J.; Pugh, D.; Skelly, T.; Stankosky, A.; Thiel, D.; Van Dantzich, M.; Wax, T.: Lifelike Computer Characters: The Persona Project at Microsoft. In: Software Agents. Menlo Park, California: AAAI Press. (1997)

[5] Koda, T. Agents with Faces: A Study on the Effects of Personification of Software Agents. Master Thesis, MIT Program in Media Arts and Sciences. (1996)

[6] Loyall, B.: Believable Agents: Building Interactive Personalities. PhD Thesis Carnegie Mellon University, Technical Report CMU-CS-97-123. (1997)

[7] Moraes, M.; Costa, A. C. R.: How Planning Becomes Improvisation? – A Constraint-Based Approach to Director Agents in Improvisational Systems. In: Advances in Artificial Intelligence, 16th Brazilian Symposium on Artificial Intelligence. LNAI 2507. Berlin: Springer. (2002)

[8] Hayes-Roth, B.; Browston, L.; Sincoff, E. Directed Improvisation by Computer Characters. Technical Report KSL-95-04 – Stanford University. (1995)

[9] Bates, J. The Nature of Characters in Interactive Worlds and The OZ Project. Technical Report CMU-CS-92-200 – Carnegie Mellon University. (1992)

[10] Perlin, K.; Golberg, A. Improv: A System for Scripting Interactive Actors in Virtual Worlds. In: Computer Graphics, vol. 29. (1996)

[11] Spolin, V. Improvisation for the Theater: A Handbook of Teaching and Directing Techniques. First Edition. Nothwestern University Press. (1963)

[12] Reverbel, Olga. Teatro: Uma Síntese em Atos e Cenas. Editora: LPM, (1987)

[13] Chacra, S.: Natureza e Sentido da Improvisação Teatral. Editora Perspectiva. (1983)

[14] Russell, S.: Analogy by Similarity. In David Helman (Ed.), Analogical Reasoning, Boston, MA: D. Reidel. (1988)

[15] Russell, S.: A Quantitative Analysis of Analogy by Similarity. In Proceedings of the Fifth National Conference on Artificial Intelligence, Philadelphia, PA: Morgan Kaufmann. (1986)

[16] Davies, T. R; Russell, S.: A Logical Approach to Reasoning by Analogy. In Proceedings of the Tenth International Joint Conference on Artificial Intelligence, Milan, Italy: Morgan Kaufmann,. (1987)

[17] Bertoletti, A. C.; Costa, A. C. R. SAGRES – A Virtual Museum. In: Museums and The Web 1999 Conference. USA: New Orleans, Louisiana. (1999)

[18] Bratman, Michael E.; Israel, David J.; Pollack, Martha E. Plans and Resource-Bounded Practical Reasoning. Computational Intelligence, 4 (4). (1988) 349-355

[19] Wooldridge, Michael. Intelligent Agents. In: Multiagent systems: A Modern Approach to Distributed Artificial Intelligence. Gerhard Weiss (editor). The MIT Press, Cambridge, Massachusetts. (1999) 27-77

Em direção a uma abordagem para separação de interesses por meio de Mineração de Aspectos e Refactoring

Vinicius C. Garcia*, Daniel Lucrédio†, Antonio F. do Prado
GOES – Grupo de Engenharia de Software
Departamento de Computação – Universidade Federal de São Carlos
Caixa Postal 676 – São Carlos, SP, Brasil
(vinicius, lucredio, prado)@dc.ufscar.br

and

Eduardo K. Piveta, Luiz C. Zancanella
Centro Tecnológico – CTC – Universidade Federal de Santa Catarina
Caixa Postal 1212 – Florianópolis, SC, Brasil
(kessler, zancanella)@inf.ufsc.br

and

Alexandre Alvaro, Eduardo S. de Almeida
Centro de Informática – Universidade Federal de Pernambuco
Caixa Postal 7851 – Recife, PE, Brasil
aa2, esa2(@cin.ufpe.br)

Abstract

This paper presents an approach to aid migration from object-oriented code, written in *Java*, to a combination of objects and aspects, using *AspectJ*. This approach supports the use of aspect mining, in order to identify possible crosscutting concerns to be implemented as aspects. The concerns, previously identified, are extracted from object-oriented code throw refactorings and encapsulated into aspects to obtain the new aspect oriented code. We present in this paper a collection of manual aspect-oriented refactorings to extract crosscutting concerns from object-oriented code.

Keywords: AspectJ, Aspect Mining, Refactoring, Aspect-Oriented Software Development

Resumo

Este artigo apresenta uma abordagem para auxiliar na migração de código orientado a objetos, escrito em *Java*, para uma combinação de objetos e aspectos usando *AspectJ*. A abordagem se apóia no uso de mineração de aspectos, de forma a identificar possíveis interesses multi-dimensionais a serem implementados como aspectos. Os interesses, previamente identificados, são extraídos do código orientado a objetos por meio de *refactorings* e encapsulados em aspectos para obter o novo código orientado a aspectos. É apresentado neste artigo uma coleção de *refactorings* orientados a aspectos para extrair interesses multi-dimensionais do código orientado a objetos.

Palavras-chaves: AspectJ, Mineração de Aspectos, *Refactoring*, Desenvolvimento de Software orientado a Aspectos

*Financiado pela Fundação de Amparo à Pesquisa do Estado da Bahia (FAPESB) - Brazil
†Financiado pela Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) - Brazil

1 Introdução

Diversas empresas são compelidas a migrar seus sistemas legados para novas linguagens ou procurar por novas formas para melhorar seus sistemas de software existentes. Isto normalmente é feito para reduzir custos de manutenção, aumentar a velocidade de desenvolvimento e melhorar a legibilidade dos sistemas.

Metodologias recentes de desenvolvimento de software utilizam a Orientação a Objetos (OO) em busca de um maior nível de reutilização, o que leva a um desenvolvimento de software mais rápido e a sistemas de melhor qualidade, facilitando futuras manutenções. O software desenvolvido usando a Programação Orientada a Objetos (POO) é mais fácil de manter porque oferece mecanismos para reduzir o acoplamento e aumentar a coesão dos módulos, melhorando a manutenibilidade.

Porém, mesmo utilizando uma metodologia OO e decompondo o sistema em um conjunto de objetos individuais, cada um representando uma funcionalidade específica, ainda existem funcionalidades que ficarão espalhadas por diferentes objetos. Isso acontece porque o paradigma OO possui algumas limitações [1], como, por exemplo, o entrelaçamento e o espalhamento de código com diferentes propósitos. Estas limitações levaram a diversas propostas na tentativa de melhorar a modularização das aplicações. Padrões de projeto [6] podem ser utilizados para atenuar essas limitações, porém não são suficientes.

O Desenvolvimento de Software Orientado a Aspectos (DSOA) [2] é um novo paradigma para o desenvolvimento de software que procura aumentar a modularidade, reduzindo os problemas com entrelaçamento e espalhamento de código, tornando a estrutura do software mais clara e de mais fácil manutenção [2]. O DSOA pode ser aplicado para a implementação de vários interesses multi-dimensionais (*crosscutting concerns*) por meio de uma unidade modular chamada de aspecto. Como exemplo destes interesses pode-se citar: interação entre objetos, segurança, persistência, distribuição e tratamento de exceções. Outro conceito introduzido pelo DSOA são os pontos de combinação (*join points*), que especificam os elementos que serão modificados pelos aspectos. Os aspectos encapsulam a implementação dos interesses multi-dimensionais e se referem a um conjunto de pontos de combinação.

Na literatura pode-se encontrar diversas propostas [4, 11, 15] para suportar a identificação de aspectos em estágios iniciais do processo de desenvolvimento de software. Embora estes aspectos a serem identificados possam ser encaixados como *"late aspects"* (ao contrário dos aspectos na fase de requisitos, denominados de *"early aspects"*), eles tornam-se necessários na definição de requisitos quando na migração de um sistema legado ou na adoção de aspectos em um projeto em andamento.

O objetivo de migrar sistemas OO para orientado a aspectos (OA) inclui, entre outros benefícios, a melhora no entendimento do sistema aumentando assim a manutenibilidade e dando uma maior garantia para a sua evolução. A identificação dos candidatos a aspectos requer primeiramente uma clara idéia de quais aspectos procura-se encontrar. Assim, um estudo sobre os aspectos genéricos ou específicos de um domínio, é um pré-requisito para a mineração de aspectos. Esse estudo pode-se iniciar a partir dos interesses multi-dimensionais como, por exemplo, a busca por pontos de combinação estáticos como, por exemplo, chamada e execução de métodos.

Em paralelo, *Refactoring* [3] é uma técnica utilizada para reestruturar código OO de uma forma disciplinada. A intenção do *refactoring* é aprimorar a a legibilidade e a compreensão do código OO. A maioria das tecnicas de *refactoring* se propõem a aumentar a modularidade e eliminar a redundância do código. Considerando que as mesmas vantagens são obtidas por meio do DSOA, parece ser natural aplicá-los no mesmo processo de desenvolvimento. O benefício de usar ambas as abordagens é considerável. *Refactoring* pode ajudar a reestruturar o código OO segundo o paradigma OA. Neste contexto, torna-se viável migrar sistemas OO para OA.

Neste artigo é proposta a utilização de técnicas de DSOA combinadas com Mineração de Aspectos e *Refactorings* para possibilitar a evolução de sistemas OO por meio da separação de interesses multi-dimensionais em aspectos. Sendo assim, o artigo está organizado da seguinte forma: na seção 2 faz-se uma apresentação sobre os conceitos do Desenvolvimento de Software Orientado a Aspectos. Na seção 3 mostram-se os avanços nas pesquisas sobre Mineração de Aspectos e os principais trabalhos nesta área. Na seção 4, além dos conceitos, definem-se alguns *refactorings* para auxiliar na extração de interesses multi-dimensionais de códigos OO. E, na seção 5, apresentam-se as considerações finais acerca dos resultados obtidos.

2 Desenvolvimento de Software Orientado a Aspectos

Na Engenharia de Software, decompor um sistema em pequenas partes é uma forma essencial de reduzir a complexidade e garantir a sua evolução. Esta decomposição resulta na *"separação de interesses"* (do inglês *separation of concerns*) e facilita o trabalho paralelo, a especialização da equipe de desenvolvimento, a localização de pontos que devem sofrer modificação e consequentemente auxilia na manutenção, testes sistemáticos e na garantia da qualidade [4].

Infelizmente, algumas funcionalidades existentes nos sistemas de software, como tratamento de erros ou segurança, são inerentemente difíceis de se decompor e isolar, reduzindo assim a legibilidade e a manutenibilidade destes sistemas.

Existem diversas formas de contornar estes problemas usando técnicas OO como a utilização de padrões de projetos [6]. Para poucas classes e/ou poucos objetos isto funcionaria adequadamente. Entretanto, a medida que o número de objetos aumenta, cresce também a quantidade de memória, do computador, necessária e o número explícito de associações entre os decoradores e os objetos decorados.

O Desenvolvimento de Software Orientado a Aspectos (DSOA), surgiu nos anos noventa como um paradigma direcionado a implementar interesses multi-dimensionais(*crosscutting concerns*) ou mais especificamente aspectos, por meio de técnicas de geração de código para combinar (*weaver*) aspectos na lógica da aplicação [2]. A separação dos interesses multi-dimensionais proporcionada pelo DSOA busca resolver os problemas de espalhamento e entrelaçamento de código encontrados em sistemas OO.

De acordo com os princípios da OA, aspectos modificam componentes de software através de mecanismos estáticos e mecanismos dinâmicos. Os mecanismos estáticos preocupam-se com a adição de estado e comportamento nas classes, enquanto que os mecanismos dinâmicos modificam a semântica destas em tempo de execução. Estes aspectos são implementados como módulos separados, de forma que fique transparente a maneira como os aspectos agem sobre os componentes funcionais, e como estes o fazem [5].

Para a definição e implementação destes mecanismos, uma abordagem OA normalmente provê, em adição às linguagens tradicionais (OO ou estruturadas) uma linguagem para a definição de aspectos e um combinador de aspectos (*aspect weaver*).

AspectJ[1] é uma linguagem de programação OA de apoio ao DSOA por meio de novas construções para implementar os interesses multi-dimensionais que utiliza a linguagem *Java* como base. Além dos mecanismos relativos a OO (classes, métodos, atributos etc) também existem mecanismos relacionados a implementação de aspectos, tais como: pontos de corte (*pointcuts*), pontos de combinação (*joinpoints*), *advices* e introduções (*introductions*).

Atualmente, como o DSOA está entrando em uma fase de inovações, novos desafios vão surgindo enquanto a tecnologia vai sendo adotada e estendida. As versões das linguagens do DSOA como *AspectJ*, as contribuições de diversos grupos de pesquisa[2] e a recente integração com servidores de aplicação como *JBOSS*[3] e *BEA's WebLogic*[4] demonstram o crescimento da popularidade do DSOA.

3 Mineração de Aspectos

Várias preocupações relativas ao uso do DSOA são levantadas, como, por exemplo, o risco de obter um código mais entrelaçado, conhecido como *"código spaghetti"*, pela não utilização correta dos conceitos da OA e o uso indiscriminado de abordagens *ad-hoc* para o projeto de sistemas, pelo fato de não haver um padrão definido para expressar a modelagem de sistemas OA. Estas preocupações levam a seguinte questão: *Quando o DSOA é necessário e quando somente a POO resolve o problema ?.*

Neste contexto, nós acreditamos que a mineração de aspectos pode ajudar a resolver esta questão.

Técnicas de mineração de software ajudam a encontrar informações valiosas no código de um sistema, tornando estas informações disponíveis aos engenheiros de software envolvidos na evolução daquele sistema. Um bom exemplo de mineração de software é a *extração de regras de negócio.*

A mineração de software é apoiada por técnicas de exploração de software [7]. Ela envolve três passos: (1) coletar dados do código fonte, (2) inferir conhecimento com base na abstração dos dados coletados, e (3) apresentar a informação usando, por exemplo, hipertexto ou outros recursos para visualização.

Para realizar a mineração de aspectos existe a necessidade de fazer um *parsing* dos programas OO e verificar os locais nos quais existam códigos duplicados, difusos ou referentes a diversas preocupações de projeto.

Considere uma implementação do padrão de projetos *Observer* utilizando a linguagem *Java* como um exemplo de mineração utilizando pontos de combinação, no qual a classe *Termometer* faz o papel de *Observer* e a classe *Celcius* faz o papel de *Subject.*

A classe *Observer*:

```
1   public abstract class Termometer{
2      private Subject subject = null;
3      private Celcius tempSource;
4      // getter and setter methods
5      public abstract void drawTemperature();
```

[1] http://eclipse.org/aspectj
[2] http://aosd.net
[3] http://www.jboss.org
[4] http://www.bea.org

```
6      public void update() { drawTemperature(); }
7   }
```

A classe *Subject*:

```
  import java.util.Vector;
1   public class Celcius implements Subject{
2     private double degrees;
3     private Vector observers = new Vector();
4     public Object getData() { return this; }
5     public double getDegrees(){ return degrees; }
6     public void setDegrees(double aDegrees){
7       degrees = aDegrees;
8       for (int i=0;i<getObservers().size();i++){ ((Observer)getObservers().elementAt(i)).update(); }
9     }
10    public void add(Observer obs) {
11      observers.addElement(obs);
12      obs.setSubject(this);
13    }
14    public void remove(Observer obs) {
15      observers.removeElement(obs);
16      obs.setSubject(null);
17    }
18    public Vector getObservers(){ return observers; }
19    Celcius(double aDegrees){ setDegrees(aDegrees); }
20  }
```

Um observador concreto, chamado de *FahrenheitTermometer*:

```
1  public class FahrenheitTermometer extends Termometer{
2    public void drawTemperature(){
3      System.out.println("Temperature in Fahrenheit:" + (1.8 * getTempSource().getDegrees())+32); }
4  }
```

E um programa de teste:

```
1  public class TestTemperatures{
2    public static void main(String a[]){
3      Celcius c = new Celcius(10);
4      Termometer termo = new CelciusTermometer();
5      Termometer termo1 = new FahrenheitTermometer();
6      termo.setTempSource(c);
7      termo1.setTempSource(c);
8      c.setDegrees(100);
9      c.add(termo);
10     c.setDegrees(200);
11     c.setDegrees(300);
12     c.add(termo1);
13     c.setDegrees(400);
14     c.setDegrees(500);
15   }
16 }
```

Uma solução seria a ferramenta de mineração verificar a ocorrência de pontos de combinação estáticos e, a partir de então, gerar aspectos de forma a rastrear estas ocorrências. Logo, para cada classe selecionada pelo usuário um aspecto é gerado, bem como pontos de corte e *advices* para interceptar chamadas, execuções de métodos, leitura e escrita de atributos ou quaisquer informações selecionadas pelo usuário. O código a seguir mostra um aspecto gerado para a classe *Celcius* sempre que uma chamada ao método `setDegrees()` é realizada.

```
1  aspect MiningCelciusSetDegrees{
2    pointcut stateChanges(Subject s): target(s) && call(void Celcius.setDegrees(..));
3      after(Subject s): stateChanges(s){
4        System.out.println("State changed at" + thisJoinPoint.getSourceLocation());
5      }
6  }
```

Ao combinar e rodar o programa `TestTemperatures` é informada na saída padrão a localização dos pontos de combinação especificados, conforme o código abaixo. Esta saída pode ser configurada de forma a gerar arquivos a serem analisados pela ferramenta de mineração. Neste exemplo, a chamada ao método `setDegrees` têm grandes chances de ser uma das chamadas que encontra-se mais espalhada pela aplicação. O engenheiro pode tentar extrair as chamadas a este método para um aspecto de forma a visualizar as mudanças de projeto e implementação resultantes desta decisão de projeto.

```
State changed at Celcius.java:34
State changed at TestTemperatures.java:13
Temperature in Celcius:200.0
State changed at TestTemperatures.java:18
Temperature in Celcius:300.0
State changed at TestTemperatures.java:19
Temperature in Celcius:400.0
Temperature in Fahrenheit:720.032
State changed at TestTemperatures.java:22
Temperature in Celcius:500.0
Temperature in Fahrenheit:900.032
State changed at TestTemperatures.java:23
```

As transformações requeridas para a geração de aspectos usando os resultados da mineração podem ser consideradas similares a *refactorings*, de forma a obter um projeto melhor. Uma *refactoring* denominada *Extract Aspect* pode ser definida de forma a tentar extrair estes fragmentos em uma única abstração.

As ferramentas existentes de mineração de aspectos [8, 9, 11] foram desenvolvidas com o objetivo de encontrar termos comuns no léxico (mineração textual) e utilizando expressões regulares baseadas em tipos. Entretanto, existem problemas com estas abordagens, que serão discutidos a seguir.

3.1 Aspect Mining Tool

A ferramenta denominada *Aspect Mining Tool* (AMT) [8] é um aplicativo que realiza a analise do código-fonte existente na tentativa de identificar e extrair código relacionado a interesses que estão espalhados através das classes do sistema. Para este trabalho, foi utilizada a versão 0.6a da ferramenta[5].

De acordo com [8], duas formas de mineração de interesses podem ser utilizadas para encontrar elementos que pertencem a um mesmo interesse multi-dimensional:

Mineração baseada em texto: A mineração de dados baseada em texto procura por padrões no código fonte utilizando os nomes de classes, métodos e atributos como base. Ela é interessante caso sejam utilizadas convenções de nomeação para os elementos do sistema. A não utilização de um padrão para tal pode dificultar no processo de identificação de interesses multi-dimensionais.

Mineração baseada em tipos: A mineração através de tipos procura pela ocorrência dos diversos tipos (i.e. classes) definidos no aplicativo a ser analisado. Ela permite encontrar pontos em um sistema nos quais o acoplamento e a coesão possam deixar a desejar. Um único módulo que utiliza alguns poucos tipos é candidato a possuir uma coesão alta e baixo acoplamento, por exemplo. A mineração baseada em tipos funciona melhor que a mineração baseada em texto quando não existem convenções de nomeação. Ela não é muito interessante quando instâncias de um mesmo tipo são utilizadas para diferentes propósitos.

Apesar dos interesses selecionados estarem dispersos nas classes da aplicação, isto não significa que sejam potenciais aspectos. Isto leva a consideração de que uma forma interessante de realizar a mineração é a utilização dos pontos de combinação disponíveis nas linguagens OA mais utilizadas. Logo, além da procura por texto e por tipo, é válida a procura através de chamadas e execuções a métodos (ou padrões de métodos), leitura e escrita de atributos, utilização de construtores etc.

Outras limitações incluem, por exemplo, o fato de que os relacionamentos de herança não são levados em consideração na mineração baseada em tipos e que não é utilizado nenhum mecanismo de inferência para sugerir ao usuário problemas de modularização. Ela apenas informa as ocorrências dos tipos ou de padrões de texto dentro das classes. Uma funcionalidade interessante seria a extração desses trechos de código em um aspecto.

3.2 FEAT

A ferramenta FEAT (*Feature Exploration and Analysis Tool*) tem por objetivo auxiliar na identificação de interesses multi-dimensionais. Para isso ela possibilita a criação de grafos de interesses [9], os quais permitem a definição de um conjunto de interesses formados por classes e fragmentos de código que são adicionados a medida que o usuário interage com o sistema. Ela é implementada como um *plugin* para o *eclipse*[6] e permite a utilização das ferramentas do ambiente enquanto tenta-se melhorar a modularização do sistema. Para esta seção, foi utilizada a versão 2.5.0 da ferramenta[7].

[5] `http://www.cs.ubc.ca/~jan/amt/`
[6] `http://eclipse.org`
[7] `http://www.cs.ubc.ca/labs/spl/projects/feat/`

A ferramenta é utilizada basicamente para localizar, descrever e analisar um interesse em um sistema implementado em *Java*. No entanto, nada impede que os conceitos utilizados pela ferramenta não possam ser estendidos para outras linguagens. É possível explicitar dependências entre os elementos em cada interesse, bem como comparar fragmentos presentes em mais de um interesse.

A idéia é produzir e analisar descrições de código que implementam interesses multidimensionais em código. O uso dos grafos de interesses permite a utilização de uma representação abstrata que pode ser mapeada para código. A vantagem desta abordagem é a utilização de uma representação de mais alto nível do que o código-fonte da aplicação. As representações geradas durante o processo de identificação e análise de interesses podem ser utilizadas como base para possíveis *refactorings* visando encapsular estes interesses em construções de uma linguagem OA.

A ferramenta utiliza *bytecodes* para extrair os relacionamentos estruturais dos elementos dos programas: classes, métodos e atributos. Para isso, o *Jikes Bytecode Toolkit* (da IBM) é utilizado para manipulação destes elementos em tempo de execução.

Entretanto, existem diversas limitações encontradas na ferramenta, que podem dificultar a sua utilização e adoção. A primeira delas refere-se aos conceitos utilizados. Ao invés de basear-se em conceitos conhecidos através da utilização de linguagens OA, o uso da ferramenta exige o domínio de uma série de conceitos adicionais [10], dificultando sua utilização.

Embora pressuponha-se que o usuário esteja ciente dos conceitos relacionados a ferramenta, seu uso não ocorre de maneira intuitiva, sendo necessários diversos passos de forma a ter resultados úteis para a análise dos interesses da aplicação. Melhorias poderiam ser realizadas de forma a utilizar as relações de interesse disponíveis nos pontos de combinação das linguages OA, como por exemplo o modelo de pontos de combinação de *AspectJ*. Tal melhoria pode facilitar na adoção em escala da ferramenta, bem como sua integração com os *plugins* do próprio *AspectJ*.

Em relação a usabilidade da FEAT, existem alguns pontos a desejar: (a) sempre que o usuário desejar anotar um trecho de código e adicioná-lo as relações de interesses existentes, é necessário mudar de perspectiva no ambiente do *eclipse*, (b) os elementos dos interesses não podem ser movimentados de um interesse para outro.

Outro fato importante é que a busca de elementos ocorre através da sintaxe dos elementos e não da sua semântica associada. Os problemas que isto pode acarretar são semelhantes aos tratados na seção anterior.

3.3 JQuery

A ferramenta *JQuery* [11] é um navegador de código *Java* desenvolvido como um *plugin* do *eclipse*. A linguagem é baseada em consultas, permitindo que o usuário defina o que deseja visualizar. Estas consultas são escritas através de predicados lógicos, semelhante a outras linguagens do paradigma lógico de programação (como *Prolog*). Nesta seção, foi utilizada a versão 2.0b da ferramenta[8].

A linguagem de consulta [12] é baseada em uma linguagem de programação lógica chamada *TyRuBa*[9], que é implementada em *Java*. O usuário pode escolher entre escrever suas próprias consultas ou utilizar uma das consultas existentes na ferramenta. As consultas trabalham por meio da manipulação de uma tabela de fatos baseada na árvore sintática abstrata do espaço de trabalho selecionado pelo usuário. Esta manipulação é feita utilizando os recursos de *parsing* do próprio *eclipse* (usando a *Java Development Tools* - JDT).

Um dos problemas relacionados a *JQuery* é dificuldade de utilização. Por exemplo, o predicado:

```
method(?M), name(?M,?name), child(?C,?M),
package(?C,?P), re_match(/^a/,?name)
```

retorna uma árvore contendo os métodos (dentro de suas respectivas classes) que começam com o caractere "a". Apesar de poder representar, de maneira sucinta, predicados como estes, é difícil elaborar estes predicados. Para tal, uma experiência de programação lógica é desejável de forma a realizar consultas mais complexas.

Em certos casos, é mais simples utilizar um protocolo de meta-objetos para obter informações sobre a estrutura de uma aplicação, de um pacote ou de um conjunto de classes. Dependendo da representação utilizada, outras linguagens de consulta seriam mais simples de serem utilizadas. Se fosse utilizada uma representação em XML do código fonte da aplicação na qual estão sendo realizadas as consultas, poderia ser utilizada *XPath* ou *XQuery*, permitindo a realização de consultas (*queries*) complexas de maneira mais simples e padronizada, utilizando uma linguagem de ampla utilização.

Apesar disto, após aprender como escrever suas próprias consultas o usuário pode beneficiar-se da possibilidade da criação de predicados arbitrários. Outro ponto importante é a existência de um conjunto de consultas pré-definidas, o que pode auxiliar na tarefa de aprendizado da linguagem de consulta.

[8] http://www.cs.ubc.ca/labs/spl/projects/jquery/
[9] http://www.cs.ubc.ca/labs/spl/projects/tyruba/

3.4 AspectJ Plugins

Um dos pontos importantes para a ampla adoção de *AspectJ* como a linguagem OA mais utilizada é a preocupação com o suporte ferramental necessário para que a linguagem possa ser efetivamente usada pela indústria de desenvolvimento de software [13, 14]. Inicialmente o suporte existia para diversos ambientes, como: *JBuilder*, *Net Beans*, *EMacs* etc. Atualmente, o foco principal é no desenvolvimento de *plugins* para o *eclipse*, um ambiente de desenvolvimento da IBM.

Estes *plugins* permitem ao usuário o desenvolvimento de software orientado a aspectos de forma semelhante ao que já é feito para projetos OO. Algumas funcionalidades providas pelos *plugins* abrangem:

- *Highlight* de palavras-chave de *AspectJ* no editor *Java*;
- Assistentes para criação de projetos usando *AspectJ* e para a criação de aspectos, análogos aos existentes para a criação de projetos OO e para a criação de classes, respectivamente;
- Visualização gráfica (através de um *Outline*) dos pontos de atuação e dos *advices* de um aspecto, bem como a visualização dos pontos de combinação que podem ser estaticamente determináveis e são afetados por um *advice*;
- Suporte a bibliotecas de aspectos (através de arquivos .jar) e de combinação de aspectos binária (usando manipulação de bytecodes com BCEL) e de múltiplos arquivos de build (usando a ferramenta *Ant* desenvolvida pelo *Apache Group*);
- Visualizador gráfico que permite verificar o grau de dispersão dos interesses multi-dimensionais ao longo das classes da aplicação; e
- Suporte a depuração de aspectos (além da depuração normal de classes provida pelo *eclipse*)

Uma das funcionalidades que auxiliam na utilização da linguagem *AspectJ* é permitir, através dos *plugins*, a visualização de quais métodos são afetados pelos *advices* definidos em um aspecto. Ao utilizar os *plugins* de *Aspectj*, têm-se informações relativas aos aspectos e aos pontos nos quais os *advices* terão efeito no código da aplicação.

O contrário também é possível. Ao examinar uma classe, podem ser visualizados em que locais os aspectos existentes na aplicação afetam estas classes. A partir de um menu de contexto, têm-se informações relativas ao aspecto e ao *advice* que afetam a classe a ser explorada.

Junto com estes *plugins*, existe ainda um visualizador que permite ao usuário constatar a dispersão do interesse sendo implementado como um aspecto.

Contudo, ainda existem alguns pontos fracos encontrados nos *plugins*. O primeiro deles remete a funcionalidade de *sintax highlight*, a qual ainda não funciona adequadamente em conjunto com os arquivos *Java*. Outra funcionalidade que perde um pouco de seu poder é o *code completion*, o qual não funciona no mesmo nível de qualidade encontrado ao manipular arquivos *Java* no ambiente. Apesar de bastante útil, o visualizador carece de melhorias em sua apresentação e otimização, de forma a melhorar a interação com o usuário.

Outro ponto a ser levado em consideração, quando na utilização dos *plugins*, diz respeito a utilização de *refactorings* nas classes afetadas por aspectos. Como os pontos de combinação em *AspectJ* podem ser especificados através de coringas ou de padrões de nomes, as *refactorings* do ambiente podem modificar os pontos de combinação afetados pelos aspectos da aplicação, tanto removendo quanto adicionando novos pontos (dependendo da *refactoring* utilizada).

Existem ainda restrições referentes a depuração: *breakpoints* podem ser colocados somente em classes (apesar de poder executar o código de aspectos usando execução passo a passo). Outra limitação, fruto da inexistência de suporte a *JSR 45*[10] é que não é possível atualmente percorrer um *advice* do tipo *around* usando execução passo a passo.

4 *Refactorings*

A utilização de *refactorings* na fase de desenvolvimento e na manutenção de software é uma prática que há muito tempo vem sendo utilizada por programadores, que sempre procuraram *"limpar"* seu código para torná-lo mais legível e para otimizar o desempenho.

O objetivo das técnicas de *refactoring* é realizar mudanças no código concluído, e operacional, para torná-lo mais legível e passível de modificações e/ou melhorar qualidades não funcionais do sistema, como, por exemplo, a performance. Para isto, a *refactoring* se apóia em técnicas bem planejadas com o objetivo

[10] *Java Specification Requests #45: Debugging Support for Other Languages,* `http://www.jcp.org/en/jsr/detail?id=45`

de diminuir o risco da inserção de erros no sistema. Desta forma, a *refactoring* pode ser definido como *"um conjunto de técnicas bem planejadas que tem o objetivo de melhorar a capacidade do sistema de se adaptar a novas necessidades e acomodar novas funcionalidades"* [3].

Refactorings são sistematicamente organizadas em catálogos, de um modo análogo aos padrões de projeto [6]. A aplicação de *refactorings* é aconselhada quando são identificados no código de um sistema trechos, denominados *"code smells"*, que requerem melhorias.

Refactoring e mineração de aspectos devem ser definidos e aplicados com algumas metas em mente como, por exemplo, na melhora da manutenibilidade do código e/ou da sua legibilidade ou extensibilidade. Conseqüentemente, antes de executar a transformação do código fonte que isolaria os candidatos a aspectos do código e o encapsularia de acordo com o paradigma OA, o engenheiro de software precisa avaliar o valor dessa migração com respeito a estas metas.

4.1 *Refactorings* para o paradigma OA

O engenheiro de software utiliza os *refactorings* para extrair os interesses que estão espalhados pelo sistema, dispersos por métodos e classes. Em algumas pesquisas [15, 16], os *refactorings* existentes são adaptados para realizar a organização de código OO, extraindo aspectos.

Os *refactorings* apresentados aqui são propostos a partir da experiência dos autores em DSOA e na realização de estudos de caso. É importante ressaltar que estes *refactorings* não abrangem todos os problemas e nem pretendem ser a solução completa para a extração de interesses multi-dimensionais do código OO, mas representam o primeiro passo para a criação de um catálogo com este objetivo.

Para facilitar a compreensão na especificação dos *refactorings* foi escolhido o mesmo formato utilizado por Fowler em [3]. Será utilizado também, assim como Fowler, um único código no exemplo de aplicação dos *refactorings*. A especificação compreende os seguintes elementos: o nome do *refactoring*; uma descrição da situação na qual ele é necessário e o que ele propõe; uma descrição da ação recomendada; pré-condições para sua aplicação, quando existir; o mecanismo de aplicação do *refactoring*; e, um código de exemplo.

O primeiro *refactoring* apresentado é o principal e está em um nível de abstração maior que os demais. Todos os outros *refactorings* relacionados com a extração de elementos ou de trechos de código do sistema OO estão inseridos neste primeiro aspecto.

Extract Aspect

Situação

Existe um interesse espalhado e entrelaçado pelo código de diversas classes e métodos do sistema. Pretende-se separar este interesse do código primário do sistema.

Ação Recomendada

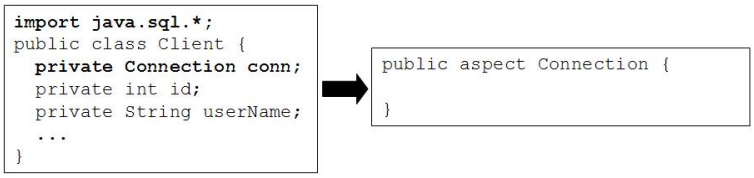
Extrair todo o código relacionado ao interesse das classes e métodos do sistema para um aspecto.

Mecanismo

- Criar um aspecto, levando em consideração o pacote em que o aspecto será criado. Caso necessário utilizar o comando *"import"*.
- Mover para o aspecto os atributos relativos ao interesse por meio do *Extract Field*. É necessário que os atributos, inicialmente, sejam declarados como públicos no aspecto até o novo código pós-aplicação deste *refactoring* ser compilado e testado.
- Mover para o aspecto os métodos relativos ao interesse por meio do *Extract Method*.
- Mover para o aspecto, como *advices*, todo trecho de código relativo ao interesse, que não seja um método por meio do *Extract Advice*.
- Modificar para *"private"* todas as declarações de membros do aspecto que devem ser visíveis somente no aspecto.
- Compilar o código e testar.

Exemplo

Como este *refactoring* está num nível de abstração maior que os demais, o exemplo irá ilustrar apenas a criação do aspecto para implementar um determinado interesse multi-dimensional, no caso o interesse de persistência.



Extract Field

Situação

Um atributo de uma classe está relacionado a um determinado interesse, e este está implementado ou foi extraído para um aspecto.

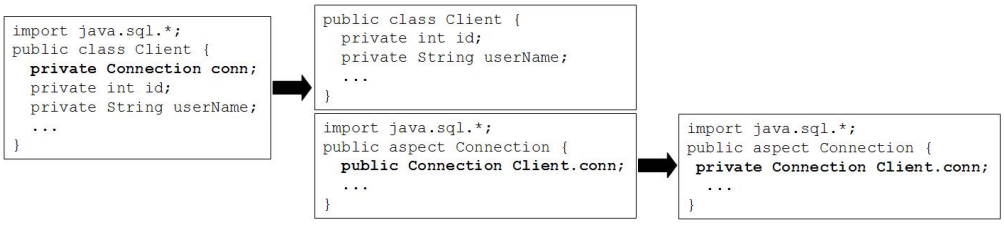
Ação Recomendada

Mover o atributo para o aspecto que implementa o interesse, como uma declaração de introdução.

Mecanismo

- Se o atributo for público, considerar o uso do *Encapsulate Field* ([3], p.206) antes da aplicação deste *refactoring*.
- Mover a declaração do atributo da classe para o aspecto, incluindo a declaração de valor inicial, caso exista.
- Adicionar o nome da classe e o "." antes do nome do atributo na declaração de introdução.
- Analisar se um novo *"import"* deve ser declarado no aspecto.
- Mudar para público a visibilidade do atributo. O valor pode voltar a ser privado assim que todo código referente ao atributo for extraído para o aspecto. Se for necessário manter o código relacionado ao atributo na classe, considerar o uso do *Self Encapsulate Field* ([3], p.146).
- Checar todos os pontos de corte com a declaração *within()* que devem ser atualizados após a aplicação do *refactoring*.
- Compilar o código e testar.

Exemplo



Extract Method

Situação

O método de uma classe está relacionado a um determinado interesse, e este está implementado, ou foi extraído para um aspecto.

Ação Recomendada

Mover o método para o aspecto que implementa o interesse, como uma declaração de introdução.

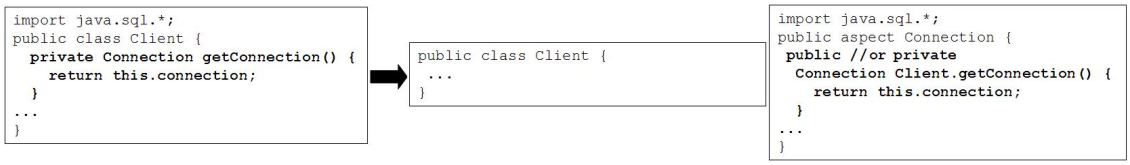
Pré-condições

Em [17], Monteiro especifica algumas pré-condições para a aplicação deste *refactoring*, principalmente se o método for público.

Mecanismo

- Mover a declaração do método da classe para o aspecto.
- Adicionar o nome da classe e o "." antes do nome do método na declaração de introdução.
- Se a visibilidade do método não for pública, mudar temporariamente para pública. Após todo código relativo ao método ser extraído para o aspecto, retornar à visibilidade ao valor original.
- Analisar se um novo *"import"* deve ser declarado no aspecto.
- Checar todos os pontos de corte com a declaração *within()* que devem ser atualizados após a aplicação do *refactoring*.
- Compilar o código e testar.

Exemplo



Extract Advice

Situação

Um trecho de código de um método está relacionado a um interesse e precisa ser extraído para o aspecto que implementa este interesse.

Ação Recomendada

Criar um ponto de corte que capture o ponto de combinação relacionado ao trecho de código, e mover o trecho de código para um *advice* apropriado.

Pré-condições

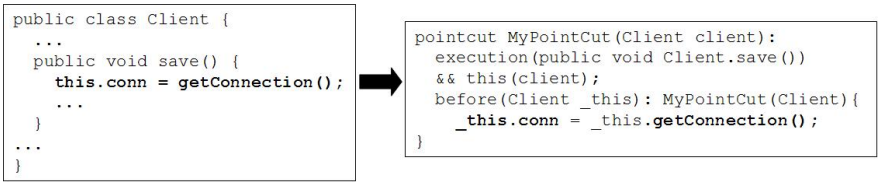
Antes de fazer a extração dos trechos de código da classe para o *advice* no aspecto, deve ser feita uma cuidadosa análise do corpo do método, para especificar o ponto de corte que irá capturar o(s) ponto(s) de combinação adequado(s). Caso o código primário não ofereça nenhum ponto de combinação adequado, outros *refactorings* podem ser aplicados para corrigir este problema. Mais detalhes sobre as pré-condições deste *refactoring* podem ser encontradas em [16] e [17] que especificam *refactorings* semelhantes a este.

Mecanismo

- Criar um ponto de corte que capture o conjunto de pontos de combinação. Se o ponto de corte já existir, então deve-se estendê-lo para incluir o ponto de combinação relacionado ao trecho extraído.
- Verificar se o ponto de corte captura todo contexto requerido pelo trecho de código e se o trecho extraído faz referência a declarações de *"this"* ou *"super"*. Os casos mais comuns inluem o uso da declaração de ponto de corte *target()* combinado com *call()*, ou o uso de *this()* combinado com *execution()*, *set()* ou *get()* [17].

- Criar o *advice* adequado para o ponto de corte, sem implementação do corpo.
- Mover o código a ser extraído para o corpo do *advice*.
- Adicionar código necessário para implementar o contexto do *advice*.
- Substitur referências às próprias variáveis *"this"* pela variável capturada no contexto do ponto de combinação.
- Analisar o código extraído em busca de referências a variáveis locais no escopo do método, incluindo parâmetros e variáveis locias. Declarações a quaisquer variáveis temporárias, usadas somente no trecho extraído, podem ser substituídas no corpo do *advice*.
- Compilar o código e testar.

Exemplo



Rename Pointcut

Situação

O nome do ponto de corte não revela seu propósito.

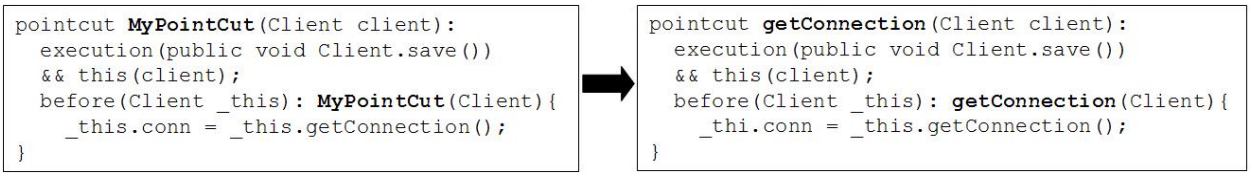
Ação Recomendada

Mudar o nome do ponto de corte.

Mecanismo

- Criar um novo ponto de corte com o novo nome. Copiar o código dos *advices* do ponto de corte com o nome antigo, para o novo ponto de corte e fazer as alterações adequadas, como, por exemplo, atualizar o nome do ponto de corte nos *advices*.
- Checar se existe algum código relacionado ao ponto de corte. Caso exista, modificar as referências ao antigo nome, para o novo.
- Remover o antigo ponto de corte.
- Compilar o código e testar.

Exemplo



Outros *refactorings*

Outros *refactorings* podem ser criados para solucionar problemas não previstos. Os trabalhos de Monteiro e Fernandes [17] e o de Hanenberg et al. [16] especificam alguns *refactorings* semelhantes e outros que complementam os que foram apresentados.

5 Conclusões

Pesquisas em Mineração de Aspectos preocupam-se com o desenvolvimento de conceitos, princípios, métodos e ferramentas para apoiar a identificação de aspectos em sistemas OO, e com o seu subsequente *refactoring* para sistemas OA.

Neste artigo foi apresentado o estado da arte na Mineração de Aspecto, indicando promissoras direções nesta área de pesquisa. As principais ferramentas existentes foram analisadas, identificando as vantagens e as desvantagens. Os resultados desta análise podem ser utilizados como base para a implementação de uma nova ferramenta ou técnica de Mineração de Aspectos em sistemas OO.

Técnicas de *refactoring* foram adaptadas para apoiar na extração de aspectos de códigos OO. Neste artigo não foi possível descrever todos os *refactorings* criados pelos autores, porém os mais importantes foram apresentados. O objetivo é fornecer um catálogo *online*, semelhante ao criado por Fowler[11], para apoiar os processos de migração de sistemas OO para OA.

O próximo passo é estudar formas de se automatizar a aplicação de alguns dos *refactorings* propostos por meio de transformações de software.

Referências

[1] Ossher, H. and Tarr, P. Using subject-oriented programming to overcome common problems in object-oriented software development/evolution. *Proceedings of 21st International Conference on Software Engineering (ICSE'99)*, 1999.

[2] Kiczales, G. et a. Aspect-Oriented Programming. *Proceedings of the 11st European Conference Object-Oriented Programming (ECOOP'97)*, 1997,

[3] Fowler, M. *Refactoring: improving the design of existing code*. Addison-Wesley, 1999.

[4] Deursen, A. van et al. Aspect Mining and Refactoring. *Proceedings of the First International Workshop on REFactoring: Achievements, Challenges, Effects (REFACE03). Held in conjunction with WCRE'2003*, 2003.

[5] Filman, R. E. and Friedman, D. P. Aspect-Oriented Programming is Quantification and Obliviousness. *Workshop on Advanced Separation of Concerns (OOPSLA'2000)*, 2000.

[6] Gamma, E. et al. *Design Patterns - Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.

[7] Moonen, L. Exploring Software Systems. *Proceedings of the 19th International Conference on Software Maintenance (ICSM'2003)*. IEEE Computer Society Press, 2003.

[8] Hannemann, J. and Kiczales, G. Overcoming the Prevalent Decomposition in Legacy Code. *Workshop on Advanced Separation of Concerns in Software Engineering (ICSE'2001)*, 2001.

[9] Robillard, M. P. and Murphy, G. C. Capturing Concern Descriptions During Program Navigation. *Workshop on Tools for Aspect-Oriented Software Development (OOPSLA 2002)*, 2002.

[10] Robillard, M. P. *Representing Concerns in Source Code. Ph.D. Thesis*. Department of Computer Science, University of British Columbia. November 2003.

[11] Janzen, D. and Volder, K. De. Navigating and Querying Code Without Getting Lost. *Proceedings of the 3rd International Conference on Aspect-Oriented Software Development (AOSD'2004)*, 2004.

[12] Volder, K. De et al. Logic Meta Programming as a Tool for Separation of Concerns. *Workshop on Aspects and Dimensions of Concerns (ECOOP'2000)*, 2000.

[13] Kersten, M. AO Tools: State of the (AspectJ) Art and Open Problems. *Workshop on Tools for Aspect-Oriented Software Development (OOPSLA'2002)*, 2002.

[14] Kersten, M. Tool Requirements for Commercial Development with AspectJ. *AOSD Workshop on Commercialization of AOSD Technology*, 2003.

[15] Iwamoto, M. and Zhao, J. Refactoring Aspect-Oriented Programs. *The 4th AOSD Modeling With UML Workshop*, 2003.

[16] Hanenberg, S. et al. Refactoring of Aspect-Oriented Software. *Net.Object Days 2003*, 2003.

[17] Monteiro, M. P. and Fernandes, J. M. Object-to-Aspect Refactorings For Feature Extraction. *Proceedings of the 3rd International Conference on Aspect-Oriented Software Development (AOSD'2004)*, ACM Press, 2004.

[11] `http://www.refactoring.com/catalog/index.html`

Beholder - Utilizando Redes Neurais MPL na Detecção de Intrusos

Fábio Bombonato
Universidade Católica de Brasília, Departamento de Ciência da Computação
Brasília, Brasil, 71966-700
bombonato@geleira.org

and

Flávia E. S. Coelho
Universidade Católica de Brasília, Departamento de Ciência da Computação
Servidores de Missão-Critica
Brasília-DF, Brasil, 71966-700
fcoelho@ucb.br

Abstract

Beholder is a system based on model that integrate SDIs and neural networks, aproaching a simple manner, but effective – to problems resolutions related to intrusion detection, moreover, obtain the advantages of use this approach, in comparing to tipical systems. The model proposed possibilities a implementation of a system anable to analyse and identify possible intrusions based on methods of anomaly detection, using a MPL neural network for detection of atacks manners – known as network scan at computer networks.
Keywords: Neural networks, IDS (Intrusion System Detection), Security.

Resumo

Beholder é um sistema baseado num modelo que integra os SDIs e as Redes Neurais, propondo uma forma simplificada – porém, efetiva – para a resolução de problemas relacionados à identificação de intrusão, além de obter as vantagens de se utilizar esta abordagem, se comparada aos sistemas usuais. O modelo proposto permitirá a implementação de um sistema capaz de analisar e identificar possíveis intrusões baseadas no método de detecção por anomalia, utilizando uma rede neural MLP para detecção de formas de ataque – conhecida como varredura em rede de computadores.
Palabras claves: Redes neurais, SDI (Sistema de Detecção de Intrusos), Segurança.

1. Introdução

É acompanhado constantemente em noticiários, informativos técnicos, *sites* especializados em segurança, o rápido crescimento dos ataques aos computadores, estes ligados ou não em rede. Este crescimento é perceptível a partir de estudos realizados por diversos órgãos como CERT [1], NBSO [2] e empresas como a Módulo [3].

Aumentar a segurança dos dados não é uma tarefa trivial. Para tal, se utilizam ferramentas com o propósito de melhorar a segurança, como *firewalls*, sistemas de filtragem de conteúdo (*proxy*), antivírus e os próprios Sistemas de Detecção de Intrusos (SDI) ou, *Intrusion Detection System* (IDS).

De acordo com Bace & Mell [4], detecção de intrusão é o processo de monitoramento de eventos que ocorrem em um sistema computacional ou em rede, analisando-os à procura de sinais de intrusão, definido como a tentativa de comprometer confiabilidade, integridade e disponibilidade ou por ultrapassar os mecanismos de segurança de um computador ou rede.

Os SDIs, normalmente, trabalham em três modos: 1) **SDI de *Host***, que se encontra enclausurado nas máquinas; 2) **SDI de Rede** (*Network-Based* IDS, ou NIDS) que analisa o que ocorre na rede e 3) **SDI híbrido** (*Hybrid IDS*) que seria a junção dos dois modos anteriores. No último modo, o monitoramento torna-se possível, através da análise e auditoria dos processos executados em cada máquina e pela captura e análise do tráfego de rede.

SDIs operam de duas formas: 1) via detecção de anomalias, baseados em **comportamento,** e 2) via identificação de assinaturas de ataques, estes baseados em **conhecimento** (também conhecido como **abuso**). A detecção por abuso refere-se à análise de algo conhecido como sendo "mau", sendo a técnica utilizada pela maioria dos sistemas de IDS. Já na detecção por anomalias, a análise é feita pela procura de padrões considerados anormais.

Conforme averiguado por [6], encontram-se algumas imperfeições, falta de eficiência ao tratar pacotes que serão analisados, alto número de falsos positivos (ocorrendo quando a ferramenta classifica uma ação como uma possível intrusão, quando na verdade trata-se de uma ação legítima) e flexibilidade limitada nos modelos vigentes de SDI. Muitos destes aspectos podem ser reduzidos ou até eliminados pelo uso das Redes Neurais Artificiais (RNA), força esta que vem alavancando diversas pesquisas, constituindo também um dos motivos para o desenvolvimento deste trabalho.

Este e outros fatos movimentam diversas pesquisas em Sistemas de Detecção de Intrusos (SDI), conforme verificado em [3] e [4], como por exemplo, o MIT, *Georgia University*, UBILAB *Laboratory*, *Research of* RST *Corporation*, além de iniciativas nacionais como o SADI (Sistema Adaptativo de Detecção de Intrusão) [6] e ACME! (*Advanced Counter-Measures Environment*) [8].

Esse artigo, portanto, irá apresentar um novo modelo de detecção de intrusos com diminuição do índice de falsos positivos, baseando-se em redes neurais.

2. IDS e Redes Neuras

Várias pesquisas na área de detecção de intrusos vêm sendo desenvolvidas, basicamente tentando diminuir a quantidade de falsos positivos e possibilitar um maior dinamismo no tratamento do conhecimento dos ataques. Estas pesquisas basicamente trabalham no desenvolvimento de novas técnicas de detecção de intrusos, sejam por abuso ou anomalia.

Por um lado, na detecção por abuso é possível encontrar algumas abordagens no desenvolvimento de IDS, segundo [7] e [9], dentre elas:

- Sistemas especialistas;
- Verificação de assinaturas;
- Redes de Petri (*Petri nets*);
- Diagramas de transição de estado.

Normalmente, a abordagem escolhida na detecção por abuso é, basicamente, a *verificação de assinaturas*, onde um sistema detecta os ataques que já são conhecidos, efetuando a comparação com uma assinatura invariável que é deixada pelos ataques.

Por outro lado, temos a detecção por anomalia, que segundo [7] e [9], incluem:

- Detecção por limiar (*threshold*);
- Estatísticas;
- Medidas baseadas em regras;
- *Data mining*;
- Algoritmos não-lineares ou classificadores (redes neurais, algoritmos genéticos, classificadores *Bayesianos)*.

Esta abordagem, muitas vezes, demonstra-se mais dinâmica às condições do ambiente. Contudo, a maior limitação para esta abordagem consiste em encontrar um limite correto sem prover alarmes falsos com muita frequência. Onde, tal fato, foi obtido pelo estudo em questão, uma vez que se desenvolveu um modelo que, através dos resultados, demonstrou ser capaz de reduzir, de forma considerável, os falsos positivos.

3. Descrição do Sistema Beholder

Dentre os diversos tipos de ataques por anomalia, foi escolhido trabalhar com os ataques de varredura em rede, por se tratar da fase inicial onde é feito um levantamento de informações relevantes que serão utilizadas no ataque propriamente dito.

O modelo apresentado trata, especificamente, da detecção de intrusos operando sobre redes TCP/IP, ou seja, será um IDS de Rede (NIDS). Procura-se detectar o comportamento de pacotes trafegando na rede, que sejam considerados anômalos. Para tal finalidade, será utilizado um módulo de redes neurais, para que este separe o tráfego considerado normal do suspeito, identificando o nível de periculosidade atribuído para o pacote analisado.

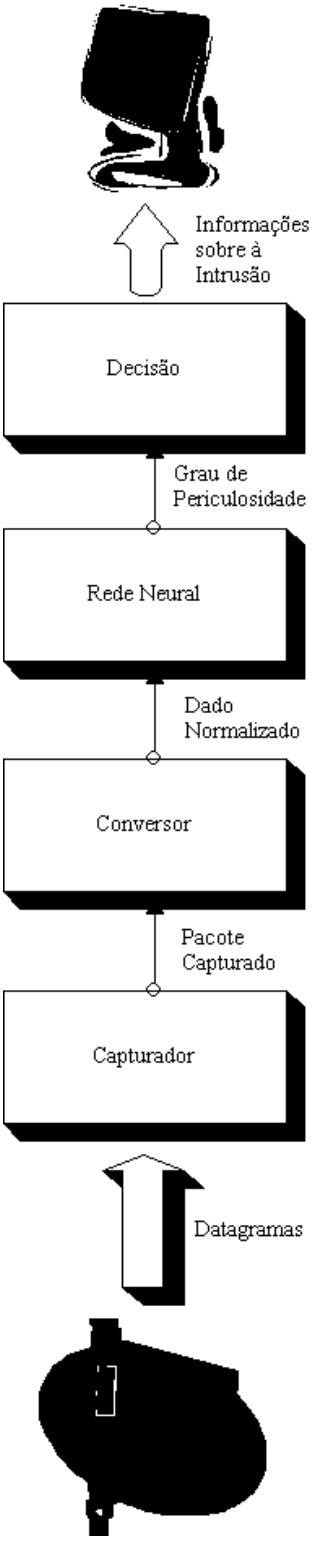


Figura 1 - Estutura geral do *Beholder*

A proposta inovadora para este sistema é a capacidade de efetuar a detecção de comportamentos anômalos utilizando redes neurais artificiais, cujo modelo aqui utilizado é, de certa forma, mais natural. Esta naturalidade é obtida, pois os dados são capturados da forma que estão sendo trafegados na rede, sendo então submetidos para a análise da RNA, evitando assim, um trabalho extra de pré-classificação e separação de fluxos.
Basicamente, o modelo geral de detecção de intrusão proposto (Figura 1), é composto de um capturador que obtém os dados da rede, um conversor que efetua a normalização destes dados recebidos da rede para então serem submetidos à rede neural artificial, ao qual, retorna um valor, indicando o grau de periculosidade referente a cada pacote analisado. Estes dados, por sua vez, são repassados ao decisor que, então, formata e apresenta ao operador conforme as duas categorias possíveis: ataque ou situação normal.

4. Fase de Aprendizagem

Para a concepção da Rede Neural utilizada, foram efetuadas algumas simulações no EasyNN e SNNS, permitindo moldar um modelo de RNA adequado para ser utilizado pela *engine Beholder*.
Os dados utilizados na simulação foram obtidos da captura de pacotes (datagramas) obtidos no barramento Ethernet, obtendo padrões normais de uso em rede e padrões considerados como ataque (obtidos de forma isolada), conforme exemplo na Tabela 1.

Protocolo	6
IP Origem	192.168.0.254
IP Destino	192.168.0.3
Porta Origem	279
Porta Destino	43309
Flags	0x14
Tamanho Header	20
Tamanho Dados	6
Tipo ICMP	0

Tabela 1 – Dados obtidos pelo Capturador

Após a coleta e seleção dos dados que serão utilizados, estes são repassados para o módulo conversor (sub-módulo simulador) que efetuará a normalização destes dados de entrada sendo convertidos para seus valores correspondentes no formato *long,* utilizado na linguagem C, permitindo assim que todos sejam formatados considerando um padrão. Feita a normalização, para o exemplo anterior, os dados seriam então apresentados no formato da Tabela 2.

Protocolo	6.0
IP O1	192.0
IP O2	168.0
IP O3	0.0
IP O4	254.0
IP D1	192.0
IP D2	168.0
IP D3	0.0
IP D4	3.0
Porta Origem	279.0
Porta Destino	43309.0
Flags	20.0
Tamanho Header	20.0
Tamanho Dados	6.0
Tipo ICMP	0.0

Tabela 2 – Dados normalizados

Com os dados normalizados foi possível efetuar a simulação que utilizou as seguintes quantidades de padrões, conforme Tabela 3.

Padrão	Quantidade
Treinamento	5577
Validação	200
Teste	18

Tabela 3 – Quantidade de padrões utilizados

O treinamento utilizado no EasyNN foi o *backpropagation* com *momentum*. Para tal, foram utilizados os seguintes parâmetros para o treinamento da rede neural utilizada:

- •Taxa de Aprendizagem: **0.30**
- •Momento (*Momentum*): **0.80**
- •Parar quando todos os erros estejam abaixo de **0.0500**

5. Rede Neural

Dadas as configurações utilizadas na fase de aprendizagem, foi possível definir a topologia da RNA para a MLP (*Multilayer Perceptron*), tendo os neurônios de entrada e saída sendo fixos, 15 padrões de entrada (conforme apresentado anteriormente) e 1 de saída (correspondendo a um valor que tende à situação de ataque ou normal). Já a camada intermediária (oculta) foi fixado para uma única camada contendo 4 neurônios.
Para esta topologia e treinamento efetuado, foram executados 44.059 ciclos de treinamento, onde a taxa de validação alcançada chegou a **99.58%** dos 200 exemplos de validação. Para os erros, foram obtidos os resultados apresentados na Tabela 4.

Tipo de Erro	Erro Obtido
Erro mínimo	0.00000
Erro médio	0.001768
Erro máximo	0.670616

Tabela 4 – Taxas de Erros

Vale ressaltar que, apesar da taxa de erro máximo estar alta, o erro médio está muito próximo do erro mínimo, o que demonstra que a rede consegue reconhecer uma grande quantidade de situações normais e de ataque. O erro máximo encontra-se elevado devido a alguns poucos padrões que não identificou de forma correta devido à similaridade com os padrões normais, conforme será visto nos resultados.

6. Resultados Obtidos

Os resultados obtidos para o ambiente proposto, podem ser resumidos abaixo:

- •Acesso Normal: Foram reconhecidos **100%** do trafego como normal. Não gerou nenhum falso positivo;
- •Ataques por Varredura: TCP SYN, *Stealth* FIN, *Xmas Tree*, *Null*, RPC: **99.5%** detectado. TCP *connect()* não foi detectado.

Os resultados apresentados foram satisfatórios, demonstrando que a utilização das redes neurais possibilita lidar com as tentativas de ataques de forma efetiva e adaptativa, visto que estas possuem uma característica interessante, conhecida como generalização. Além disso, foi constatada a grande diminuição dos falsos positivos, se comparado aos modelos usuais.
Também é importante salientar que não foi possível detectar a técnica de varredura conhecida como TCP *connect()*, tal fato já era esperado à medida que o projeto vinha sendo desenvolvido e pelo trabalho sobre os padrões de treinamento. Isto ocorre devido a grande semelhança entre os padrões relacionados ao ataque TCP *connect()* e uma situação normal, sendo praticamente idênticos.

7. Conclusões

As redes neurais artificiais demonstraram ser capazes de efetuar a detecção de intrusão baseada em anomalia, mas precisamente, para as técnicas de varredura em rede de computadores utilizando a rede neural MLP. A utilização das RNA no contexto de SDIs é considerada uma área relativamente nova, sendo necessários mais experimentos e melhoramentos para que esta abordagem se torne mais efetiva e possa ser utilizado em grande escala, ressaltando que este trabalho mostrou que esta é uma possibilidade real.

Referências

[1] Bace, R., Mell, P. NIST Special Publication on Intrusion Detection Systems. (2001)

[2] Martino, S. A Mobile Agent Approach to Intusion Detection. *Joint Research Center Institute for Systems*, Informatics and Safety, Italy. (1999).

[3] Cannady, J. Artificial Neural Networks for Misuse Detection, *School of Computer and Information Sciences*, Nova Southeastern University, (1998)

[4] Philippe Jean, Application of Neural Networks to Intrusion Detection, *Information Security Reading Room*, SANS Institute, (2001).

[5] Tavares, D. M., Castejon, E. F., Rossi, G. B. ACME! (Advanced Counter-Measures Environment), *Monografia de Projeto Final apresentada ao Departamento de Ciências da Computação*, Instituto de Biociências Letras e Ciências Exatas – UNESP, São José do Rio Preto, (1999).

[6] Oliveira, C. B. Reconhecimento de Padrões com Auxílio à Detecção de Intrusão em Redes de Computadores, *Tese de Mestrado*, Universidade Católica de Brasília, Brasília, 2001.

Optimización Multiobjetivo para la Ubicación de Locutorios de Cabinas Telefónicas

Nilton Amarilla
Universidad Nacional de Asunción
Campus Universitario de San Lorenzo, Paraguay
Casilla de Correos 1439
dmantest@copaco.com.py

Carlos D. Almeida
Universidad Nacional de Asunción
Campus Universitario de San Lorenzo, Paraguay
Casilla de Correos 1439
cdad@ieee.org

Benjamín Barán
Centro Nacional de Computación
Universidad Nacional de Asunción
Campus Universitario de San Lorenzo, Paraguay
Casilla de Correos 1439
bbaran@cba.com.py

Resumen

El problema de localización de centros proveedores de servicios (*facilities*) sobre un área determinada es un problema *NP-hard,* ampliamente estudiado en las literaturas de Investigación de Operaciones. El problema considera un conjunto de lugares factibles en los cuales se puede abrir un centro proveedor de servicio; tales como sucursales de tiendas comerciales, locutorios de cabinas telefónicas, silos, etc. La apertura de estos centros implica un costo de inversión y una presunta ganancia futura que se desean optimizar. A diferencia de las herramientas hasta ahora conocidas para dar solución a problemas de esta naturaleza, el presente trabajo propone la utilización de Algoritmos Evolutivos Multiobjetivos para la ubicación óptima de locutorios de cabinas telefónicas, garantizando la obtención de soluciones óptimas de varios objetivos simultáneos, a diferencia de los métodos mono-objetivo tradicionales. Este trabajo proporciona una herramienta válida en la obtención de propuestas óptimas de solución, teniendo en cuenta la rapidez con que se pueden encontrar estas soluciones de alta calidad. Resultados experimentales con la ubicación de locutorios de cabinas telefónicas para la ciudad de Asunción validan la presente propuesta.

Palabras claves: Ubicación de Locutorios de cabinas telefónicas, Optimización Multiobjetivo, Algoritmos Evolutivos, Pareto.

Abstract

The facility location problem on a specific area is a NP-Hard problem, widely studied in the Operations Research literature. The problem considers feasible places in which it is possible to locate a facility, like a branch of commercial store, communication centers, warehouses, etc. To open facilities involves cost and revenues that are subject to optimization. In contrast to other known and available tools, this work proposes the use of Multiobjetive Evolutionary Algorithms to optimize the location of communication centers, guarantying to achieve the best-compromised solutions, not only for one specific objective, like traditional methods, but for all considered objectives. The present work proved to be a useful tool to calculate optimal solutions, taking into account the quickness to find solutions of high quality. Experimental results with the location of communication centers for the city of Asuncion supports this proposal.

Keywords: Location of communication center, Multiobjetive Optimization, Evolutionary Algorithm, Pareto.

1 Introducción

La apertura del mercado de las telecomunicaciones, en lo que respecta a la explotación de los servicios de telefonía pública, ha producido un auge en la instalación de locutorios de cabinas telefónicas en el Paraguay. En los inicios de la apertura del mercado, debido a la carencia de servicios de esta naturaleza, las zonas comerciales y populosas se constituían en sitios aptos y favorables para la ubicación de estos locutorios y predominando de esta forma criterios meramente intuitivos al momento de escoger los sitios para ubicar estos locutorios. Sin embargo, en la actualidad el mercado se halla saturado de locutorios de cabinas telefónicas en las zonas comerciales, con la consecuente caída de la rentabilidad en la mayoría de los casos. Como resultado del crecimiento desorientado de estos servicios, desde el punto de vista de la demanda, se tiene un mercado congestionado en donde los proveedores de servicio de este rubro producen un desbordamiento por sobre la demanda en algunas zonas, sin que ello implique la inexistencia de demanda insatisfecha o una disminución de la misma en otras áreas.

El presente trabajo, pretende dar solución a la problemática planteada buscando crear una herramienta que permita ubicar un número determinado de locutorios de cabinas telefónicas, teniendo en cuenta la metodología empleada para la Optimización Multiobjetivo en la Planificación de Centrales Telefónicas [1], de forma a minimizar el costo de inversión inicial, maximizar las ganancias mensuales respectivas y minimizar el tiempo de recupero de la inversión. Todo esto, basado en datos de densidad poblacional, demanda de tráfico telefónico, densidad telefónica, densidad de locutorios de cabinas telefónicas y costos de infraestructura requerida.

Ya en la década de los años sesenta, los problemas de localización de recursos o centros proveedores de servicios fueron ampliamente estudiados en la literatura de Investigación de Operaciones; tales como programas heurísticos para localización de depósitos y optimización de redes [17], la búsqueda de soluciones enteras para programas lineales [18], localización de fábricas [4] y otros trabajos que establecen modelos para la localización y número de fábricas [15, 16]. En la actualidad estos problemas son conocidos como *facility location problems* [6], y son caracterizados por los siguientes cuatro elementos:

- un conjunto de sitios, donde es factible ubicar centros proveedores de servicio;
- un conjunto de puntos de demanda (clientes), que deben ser asignados para recibir servicio;
- una lista de requerimientos a ser cumplidos por los centros proveedores de servicio y los clientes que reciben ese servicio, y
- una función que asocie a cada conjunto de centros proveedores de servicio los costos incurridos por habilitación de los centros y asignación de clientes.

En los problemas de este tipo, el objetivo es encontrar un conjunto de centros proveedores a ser habilitados, de forma a optimizar una función de costo dada. Dentro de este contexto se han desarrollado una variedad de trabajos que dan solución al problema [7, 14], en su mayoría basados en algoritmos de aproximación (ρ-*approximation algorithm*) [9, 11, 13], que son algoritmos de tiempo polinómico que encuentran una solución factible con el valor de la función objetivo, dentro de un factor ρ del óptimo [9]. Como se puede notar, en todos los casos citados se da solución al problema, optimizando un único objetivo, específicamente, minimizando costos.

A diferencia de los trabajos referenciados, este trabajo propone resolver el problema de localización optimizando múltiples objetivos. En tal sentido, se ubicarán un conjunto de locutorios de cabinas telefónicas, considerando simultáneamente tres objetivos:

1. el costo inicial de ubicar una cantidad calculada de locutorios de cabinas telefónicas;
2. la ganancia mensual de cada uno de los locutorios ubicados, y
3. el tiempo de recupero del capital invertido en el conjunto de locutorios.

La optimización simultánea de estos objetivos en la búsqueda de soluciones se realizará utilizando Algoritmos Evolutivos Multiobjetivos que permitan encontrar soluciones al problema de referencia, optimizando todos los objetivos propuestos al mismo tiempo. En contrapartida a una solución mono-objetivo, la solución que se busca es un conjunto de soluciones Pareto que contiene a todas las soluciones de compromiso, obtenidas al considerar simultáneamente todas las funciones objetivos. En consecuencia, el responsable de la toma de decisiones obtiene un conjunto de posibilidades óptimas, en el sentido Pareto, para elegir la solución que mejor se adecue a sus necesidades. Una importante ventaja de esta metodología es que los tiempos de corridas de estos algoritmos evolutivos son considerablemente más cortos que los requeridos para calcular un conjunto similar de soluciones Pareto, utilizando repetidamente otros métodos mono-objetivos. El presente trabajo, propone la optimización del problema planteado utilizando un Algoritmo Evolutivo Multiobjetivo basado en el *Strength Pareto Evolutionary Algorithm 2* - SPEA2 [22].

Este trabajo está organizado de la siguiente manera: En la sección 2 se formula matemáticamente el problema, exponiendo algunos conceptos relativos a la optimización multiobjetivo, el método utilizado para ubicar los locutorios de cabinas telefónicas, y el problema de prueba. En la sección 3, se describe el Algoritmo Evolutivo Multiobjetivo propuesto. En la sección 4 se presentan los resultados experimentales obtenidos y su interpretación. Finalmente, se concluye el trabajo en la sección 5.

2 Formulación Matemática del Problema

En esta sección se definen algunos conceptos relativos a la optimización Multiobjetivo, se resume el procedimiento realizado para encontrar estas soluciones y se presenta el problema de prueba.

2.1 Optimización Multiobjetivo

El problema de optimización Multiobjetivo tratado en este trabajo se define de la siguiente forma [1, 3, 5]:

Optimizar $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), f_3(\boldsymbol{x}))$ **(1)**

donde $\boldsymbol{x} = (x_1, x_2, \ldots, x_i, \ldots, x_n) \in \boldsymbol{X} \subset \boldsymbol{N}^n$ representa el vector de decisión;

$\boldsymbol{y} = (y_1, y_2, y_3) = \boldsymbol{f}(\boldsymbol{x}) \in \boldsymbol{Y} \subset \boldsymbol{N}^3$ representa el vector de objetivos;

$f_1(\boldsymbol{x})$ ... costo inicial de ubicar un número calculado de locutorios;

$f_2(\boldsymbol{x})$ ... suma de las ganancias mensuales de cada uno de los locutorios ubicados;

$f_3(\boldsymbol{x})$ ... tiempo de recupero del capital invertido en el conjunto de locutorios;

$\boldsymbol{X}$ ...espacio de variables de decisión;

$\boldsymbol{Y}$ ...espacio de objetivos;

n ... número máximo de locutorios de cabinas telefónicas;

m ... número máximo de cuadrículas en que se divide el área en estudio;

x_i ... designa la ubicación de un locutorio dentro del área en estudio $(0 \leq x_i \leq m)$;

En un contexto multiobjetivo [3] se dice que un vector objetivo $\boldsymbol{y}$ domina a otro $\boldsymbol{y'}$ sí por los menos es tan bueno como aquel, o mejor que aquel, en todos los objetivos.

Una solución $\boldsymbol{x}^* \in \boldsymbol{X}$ es Pareto óptima si no existe otra $\boldsymbol{x} \in \boldsymbol{X}$ tal que $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$ domine a $\mathbf{y}^* = \boldsymbol{f}(\boldsymbol{x}^*)$. El conjunto de todas las soluciones Pareto óptimas es denominado conjunto Pareto óptimo $\boldsymbol{PO}$ $(\boldsymbol{PO} \subset \boldsymbol{X})$, y su imagen, Frente Pareto $\boldsymbol{FP}$ $(\boldsymbol{FP} \subset \boldsymbol{Y})$.

2.2 Ubicación de Locutorios de Cabinas Telefónicas y Problema de Prueba

El problema de la ubicación óptima de locutorios de cabinas telefónicas consiste en encontrar el número óptimo de locutorios de cabinas telefónicas y la mejor ubicación de las mismas, en un área de estudio (una ciudad determinada, Asunción para este trabajo), de forma a minimizar el costo integral de inversión inicial, maximizar las ganancias mensuales del conjunto y minimizar el tiempo de recupero del capital invertido.

El área de la ciudad a ser atendida se divide en m cuadrículas típicamente de 500 m de lado. A cada una de estas cuadrículas se le asigna un valor de fila y columna, conformando una matriz de datos. A cada elemento de esta matriz se asocian cinco valores: *Población*, que es la cantidad de habitantes existente en cada cuadrícula; *Teléfonos,* que es la cantidad de teléfonos residenciales existentes; *Cabinas,* que es la cantidad de locutorios de cabinas telefónicas que ya existen; *Zonas Comerciales*, que indica la naturaleza comercial o cultural de cada cuadrícula; y *Costo del Terreno* (por metro cuadrado). Los datos citados se obtuvieron a partir de datos oficiales disponibles sobre el área en estudio; que para el presente trabajo, será la ciudad de Asunción, capital de la República del Paraguay [10].

Debido a que el plano del área en estudio tiene en general una figura geométrica irregular, muchas cuadrículas caen fuera de los límites de la ciudad o en zonas no habitadas, con ríos, lagos o montañas. Por lo tanto, utilizando técnicas de matrices esparzas, a todas las cuadrículas que quedan fuera de la ciudad se les asigna un indicador de cuadrícula no válida (*flag*) y no se las cuenta entre las m cuadrículas válidas.

De esta forma, obtenemos una matriz $\boldsymbol{M} \in \boldsymbol{N}^{mx7}$ con una fila por cada una de las m cuadrículas válidas y 7 columnas con información por cuadrícula, de:

- 1ª columna: fila para su ubicación en el mapa (coordenada de *abscisa*);
- 2ª columna: columna para su ubicación en el mapa (coordenada de *ordenada*);

- 3ª columna: población actual (dato utilizado para estimar demanda);
- 4ª columna: cantidad de teléfonos residenciales (dato utilizado para estimar requerimiento de líneas);
- 5ª columna: cantidad de locutorios de cabinas telefónicas (competencia de otros proveedores);
- 6ª columna: denotación de zona (1 indica zona comercial, 0 zona residencial) para estimación de tráfico;
- 7ª columna: costo del terreno (utilizado para estimar un costo de alquiler).

La estimación de requerimiento de líneas para cada una de las cuadrículas es obtenida considerando la densidad de teléfonos residenciales, la densidad de locutorios de cabinas telefónicas ya existentes, la densidad poblacional, y la denotación de la zona para establecer una estimación de tráfico acorde, siendo ésta igual a 0,03 Erlang por abonado para las zonas residenciales e igual a 0,05 Erlang por abonado para las zonas comerciales o culturales [19]. De ésta forma el tráfico cursado medido en Erlang por cada cuadrícula es obtenida por la siguiente expresión:

$$a = 0{,}1.pob.traf \tag{2}$$

donde:

0,1 ...índice de penetración estimada de demanda;

pob ...población de cada cuadrícula;

$$traf = \begin{cases} 0{,}03 & \text{si la zona es residencial} \\ 0{,}05 & \text{si la zona es comercial o cultural} \end{cases}$$

Los valores de tráfico cursado, calculados para cada cuadrícula según la ecuación (2), son sometidas a la fórmula de Erlang B dada en la ecuación (3) [13], con un grado de servicio del 10% para estimar la cantidad de líneas telefónicas necesarias.

$$B = \frac{\frac{a^N}{N!}}{\left[1 + \frac{a}{1!} + \frac{a^2}{2!} + \cdots + \frac{a^N}{N!}\right]} \tag{3}$$

donde:

B ...grado de servicio;

N ...número de líneas necesarias para cursar el tráfico *a* estimado.

Los datos de cantidad de líneas necesarias para cada una de las cuadrículas sobre el área de estudio son almacenados en una matriz denominada matriz *Escasez*, a partir de la cual se genera la población de soluciones iniciales para el algoritmo evolutivo empleado, el cual se lleva a cabo con la implementación del algoritmo heurístico mostrado en el pseudocódigo 1 de la sección 3.1.

El costo de implementación del número total de locutorios de cabinas telefónicas, distribuidas en toda el área de estudio, es calculado de la siguiente forma [1]:

$$y_1(\boldsymbol{x}) = \sum_{i=1}^{nl} \sum_{t=1}^{m} c_{it} \,.\, h_{it} \tag{4}$$

$$h_{it} = \begin{cases} 1 & \text{si la demanda del sitio } x_t \text{ es asignada al locutorio } x_i \\ 0 & \text{caso contrario} \end{cases}$$

donde:

nl ... número de locutorios que contiene la solución $\boldsymbol{x}$, en el caso de este trabajo $6 \le nl \le nmax$;

nmax ... número máximo de locutorios a ubicar en el área de estudio;

c_{it} representa el costo por locutorio y está estimado en este trabajo por la siguiente fórmula:

$$c_{it} = (1{,}9 + 0{,}8.l_t + ct_t).w_i \tag{5}$$

donde:

l_t ... número de líneas telefónicas necesarias en la cuadrícula x_t donde se ubica el locutorio x_i (de la matriz *Escasez*);

ct_t ... costo de alquiler de la cuadrícula x_t donde se ubica el locutorio x_i; y

$$w_i = \begin{cases} 1 & \text{si se ubica un locutorio en el sitio } x_i \\ 0 & \text{caso contrario} \end{cases}$$

Cabe mencionar que la expresión de costo (5) fue obtenida haciendo proporcional el crecimiento del costo con el número de líneas o cabinas telefónicas necesarias en el locutorio considerado. Este costo de implementación considera los gastos en concepto de: mobiliario, licencia para explotar el servicio de telefonía pública, costo de líneas telefónicas y equipos para cada cabina telefónica. El costo de alquiler $ct(\mathbf{x})$ fue considerado en forma independiente, dado que su valor no se halla afectado por el número de líneas necesarias en el locutorio.

Teniendo en cuenta que uno de los objetivos del problema es maximizar las ganancias mensuales mientras que los otros dos objetivos deben ser minimizados, se optó por minimizar las ganancias negativas a los efectos de simplificar el código implementado, planteando el mismo como un problema de minimización enteramente. De este modo, la ganancia total mensual de los locutorios de cabinas telefónicas ubicados es calculada de la siguiente forma:

$$y_2(\mathbf{x}) = (-1).\sum_{j=1}^{nl} (k.p_j.e_j - 160.z) \qquad \textbf{(6)}$$

donde:

k es una constante que considera el producto del índice de penetración de la población, el porcentaje de ganancia sobre el ingreso total, la constante de proporcionalidad de acuerdo a las necesidades de líneas de cada cuadrícula y la cantidad promedio de días laborables en el mes.

p_j ... población correspondiente a la cuadrícula denotada por el elemento j del vector $\boldsymbol{x}$;

e_j ... escasez de la cuadrícula o número de líneas necesarias para la cuadrícula considerada;

z ... factor que determina el costo operativo dependiendo si la zona es comercial o residencial;

$$z = \begin{cases} 1 & \text{si la cuadrícula es comercial} \\ 0{,}8 & \text{si la cuadrícula es residencial} \end{cases}$$

El tiempo de recupero del capital invertido es calculado según la siguiente expresión:

$$y_3(\mathbf{x}) = \min(r) \qquad \textbf{(7)}$$

donde r puede ser calculado a partir de la siguiente fórmula [20]:

$$A = P.\frac{(1+in)^r.in}{(1+in)^r - 1} \qquad \textbf{(8)}$$

siendo el plazo de recupero de la inversión:

$$r = \frac{\ln(A) - \ln(A - P.in)}{\ln(1+in)} \qquad \textbf{(9)}$$

donde:

A … amortización del capital (es el monto de capital a devolver mes a mes);

P … valor presente del capital (es el capital tomado en préstamo para realizar la inversión);

in … interés mensual, y

r … plazo en meses para terminar de devolver el capital tomado en préstamo.

A los efectos de evitar estimaciones muy ajustadas para los plazos de recupero del capital invertido, el costo de inversión calculado es redondeado por exceso a múltiplos de 5.000 US$ antes de someterlo a la ecuación (9) y una vez obtenido r este es nuevamente redondeado a múltiplos de 6 seis meses.

2.3 Problema de Prueba

Como problema de prueba se escogió ubicar los locutorios de cabinas telefónicas en la ciudad de Asunción, dada la disponibilidad de datos para la misma [10]. La Figura 1.a representa el plano cuadriculado de la ciudad de Asunción, con los contornos indicando los elementos válidos de la matriz. Para este ejemplo, existen m = 499 cuadrículas válidas, numeradas como muestra la figura 1.b.

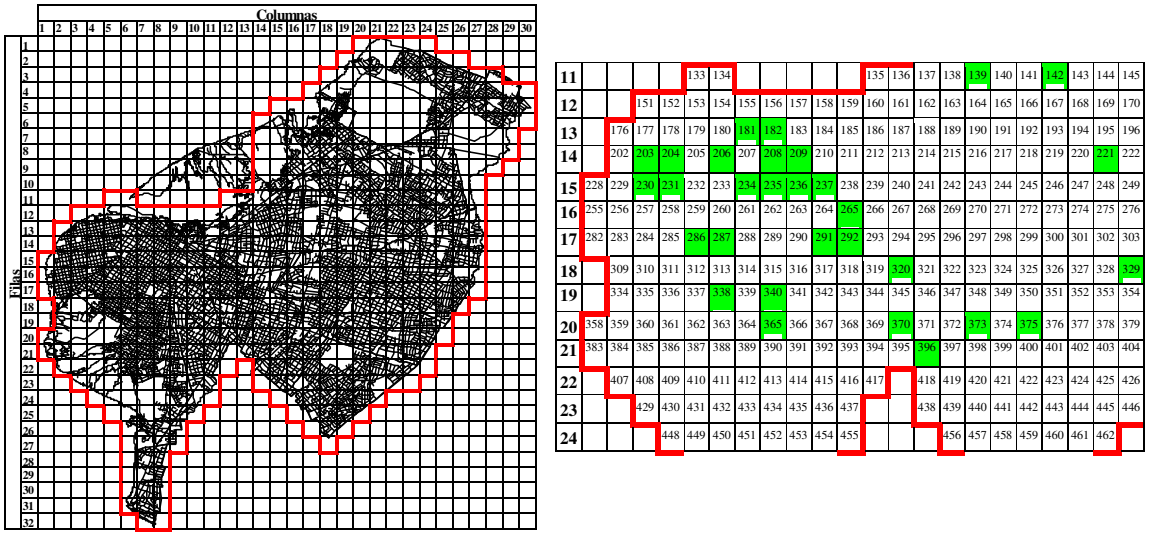


Figura 1: **a)** Delimitación de las cuadrículas válidas sobre el plano de Asunción. Se eliminan aquellas cuadrículas que caen fuera de los límites de la ciudad. **b)** Porcion del área de estudio que contiene las cuadrículas numeradas con un ejemplo de ubicación de 30 locutorios de cabinas telefónicas.

El vector de decisión para el ejemplo mostrado en la figura 1.b, al adoptar un número máximo de $nmax$ = 30 locutorios de cabinas telefónicas, corresponde a la solución efectivamente implementada en el año 2002, el cual es:

$\boldsymbol{x}$= *(139 142 181 182 203 204 206 208 209 221 230 231 234 235 236 237 265 286 287 291 292 320 329 338 340 365 370 373 375 396)*

Atendiendo el área de influencia de cada cuadrícula, que desde el punto de vista del usuario es extenso y desde el punto de vista de la demanda de tráfico es pequeño, y a los efectos de simplificar, se admitirá ubicar únicamente un locutorio por cada cuadrícula, el cual contemplará la necesidad total de líneas telefónicas de la misma. Cada cuadrícula será denotada en adelante x_t, atendiendo la condición $1 \le x_t \le m$.

En consecuencia, el problema principal a ser resuelto consiste en encontrar la cantidad de locutorios de cabinas telefónicas y la ubicación óptima de las mismas en el área de estudio, de la cual se conocen todos los datos relativos a la matriz **M** arriba definida. Si existen m sitios posibles, existen claramente 2^m alternativas de ubicación de locutorios. Aún, si se restringe la atención para ubicar nc locutorios de cabinas telefónicas en m sitios, el número de alternativas de ubicación es:

$$\binom{m}{nc} = \frac{m!}{(m-nc)!nc!} \qquad \textbf{(10)}$$

En el ejemplo de la Figura 1.b, para m=499 cuadrículas válidas y nc=30 centrales, existen unas $1{,}3\times10^{48}$ alternativas.

El problema propuesto en el presente trabajo permite encontrar soluciones Pareto que minimicen los costos de inversión y el tiempo de recupero del capital invertido y maximicen las ganancias mensuales sobre un conjunto de alternativas de ubicación, considerando los diferentes valores posibles del número de locutorios ($nmin \le nc \le nmax$).

El espacio de búsqueda del problema propuesto es, entonces:

$$\sum_{i=nmin}^{nc} \binom{m}{i} \quad \textbf{(11)}$$

que equivale a unas $1{,}45\times10^{48}$ alternativas.

En otras palabras, el método a ser utilizado en el presente trabajo debe posibilitar la obtención de un conjunto de soluciones Pareto óptimas, estableciendo la cantidad y la ubicación óptima de estos locutorios de cabinas telefónicas.

3 Algoritmo Evolutivo Propuesto

El algoritmo evolutivo propuesto es el SPEA 2, cuyo desempeño en la búsqueda de soluciones se caracteriza por la obtención de soluciones Pareto óptimas y la diversidad de las mismas sobre el Frente Pareto. Este algoritmo utiliza una estrategia de asignación de *fitness* que incorpora información de densidad a fin de evitar la pérdida de posibles soluciones óptimas [22]. El operador de truncamiento elimina aquellos individuos que están muy pegados unos a otros de forma a no perder puntos valiosos de la frontera y asegurar de ésta forma que las soluciones encontradas en el frente Pareto sean regularmente distribuidas. El proceso de encontrar los individuos no dominados en el archivo y la población está basado en el concepto de dominancia Pareto. Cada vez que un individuo no dominado es encontrado, el mismo es comparado con los no dominados ya existentes en el archivo, y si el mismo es una solución, el individuo hallado es insertado en el archivo. Para esclarecer el procedimiento de aplicación del SPEA 2 en la ubicación de locutorios de cabinas telefónicas, a continuación se presenta un esquema de utilización del referido algoritmo.

3.1 Representación de Soluciones y Población Inicial

Para la aplicación de los Algoritmos Evolutivos Multiobjetivos propuestos en el problema de prueba, cada individuo $x = (x_1, x_2, ..., x_i, ..., x_{nc})$ fue codificado usando un arreglo de números enteros x_i, tal que $0 \le x_i \le m$ $(m=499)$. En la figura 1, donde se representa el plano cuadriculado de Asunción, se puede apreciar los 499 valores no nulos de la matriz utilizada para los cálculos de costos de inversión, cálculos de ganancias mensuales y estimación del plazo de recupero del capital de cada vector de decisión. La población inicial, cuyo tamaño se denotará como *"nind"* (número de individuos), es generada por un algoritmo heurístico de inicialización detallado en el Pseudocódigo 1, en donde "*nmax*" indica el número máximo de centrales para cada vector de decisión. Este algoritmo genera una población inicial en forma inteligente de manera a obtener individuos que se aproximen razonablemente al conjunto de soluciones Pareto óptimas buscadas, minimizando de esta forma los tiempos de corridas. Para cada individuo de la población, se realiza un sorteo para saber cuantos locutorios de cabinas telefónicas tendrá esa solución. El algoritmo heurístico de inicio de la población se describe a continuación.

Algoritmo Heurístico de Inicialización de la Población Inicial.
Leer parámetros: *posibles, nind, nmax, nmin*
Ordenar matriz de *posibles* de acuerdo al número de líneas telefónicas necesarias
Para *i*=1 hasta *nind*
    Generar un número aleatorio *N* entre *nmin* y *nmax*
    Si *N*< *nmin*
        N=*nmin* (el Nro. Mínimo de locutorios a ubicar para la solución inicial será *nmin*)
    Fin si
    Para *j*=1 hasta *N*
        Elegir una ubicación *indice* de la matriz de *posibles*
        inisolu (i, j)= *posibles (indice)*
    Fin Para
    Si *N* < *nmax*
        inisolu (i, j) =0 para todo *i* que no contiene un locutorio (esto es, $N+1 \le i \le nmax$)
    Fin Si
    Inicializar *N*=0
Fin Para
inisolu=Matriz de soluciones iniciales (dimensión *nind* x *nmax*)
Ordenar *inisolu* por columna

Pseudocódigo 1: Algoritmo Heurístico de generación de la población inicial.

3.2 Evaluación de Soluciones y Función Fitness

En la evaluación de la función *fitness* se utilizaron los conceptos de dominancia Pareto definidos en la sección 2.1 en un contexto de minimización de funciones objetivos. De ésta forma, cada vector de decisión es comparado con otro a través de las funciones objetivo de dichos vectores, de tal forma a determinar si un individuo *i* domina a otro individuo *j*. La función *fitness* fue implementada conforme a lo especificado por el SPEA2 de Zitzler [22], detallado en el Pseudocódigo 2.

Los valores de *fitness* calculados mediante esta función son utilizados en la selección de los individuos que pasarán a formar parte del archivo que contiene a los mejores individuos de la población. El referido algoritmo asigna a los individuos no dominados un *fitness* menor a 1, en cuanto que a los individuos dominados se les asigna un *fitness* mayor o igual a 1, con lo que todos los individuos tienen diferentes valores de *fitness*.

3.3 Selección

Se denomina como *selección del ambiente* [22] a la acción de completar con los mejores individuos de cada generación una población externa denominada archivo. El tamaño del archivo es fijo y no varía durante las corridas del algoritmo. Inicialmente, todos los individuos no dominados, cuyos fitness son menores que uno, son copiados al archivo de la siguiente generación $\overline{P}_{t+1} = \{i \mid i \in P_t + \overline{P}_t \wedge F(i) < 1\}$. Si la cantidad de individuos no dominados es igual al tamaño establecido para dicho archivo $(|\overline{P}_{t+1}|=\overline{N})$, el paso de *selección del ambiente* está completo. Caso contrario, existen dos posibilidades:

1) la cantidad de individuos no dominados es menor que el tamaño establecido para el archivo $(|\overline{P}_{t+1}|<\overline{N})$, o

2) la cantidad de no dominados es mayor que el tamaño fijado para el archivo $(|\overline{P}_{t+1}|>\overline{N})$.

En el primer caso, se completa el archivo con los mejores $(\overline{N}-|\overline{P}_{t+1}|)$ individuos dominados en el archivo y la población de la generación anterior *t*. Esto es implementado ordenando el multiconjunto $P_t+\overline{P}_t$ de acuerdo a los valores de fitness y copiando a $\overline{P}_{t+1}$ los primeros $\overline{N}-|\overline{P}_{t+1}|$ individuos *i* con fitness $F(i) \geq 1$. En el segundo caso, cuando el tamaño del conjunto de no dominados es mayor a $\overline{N}$, un operador de truncamiento remueve iterativamente los individuos de $\overline{P}_{t+1}$ hasta que el conjunto de no dominados sea igual al tamaño establecido para el archivo $|\overline{P}_{t+1}|=\overline{N}$. Este operador de truncamiento garantiza que puntos valiosos de la frontera no sean perdidos, y lo realiza de la siguiente forma: el individuo que tiene la menor distancia euclidiana a otro individuo es desechado en cada iteración. En caso de igualdad con otros individuos, se desempata considerando la segunda menor distancia del individuo a ser removido, y así sucesivamente.

3.4 Pseudocódigo del Algoritmo Evolutivo Multiobjetivo Propuesto SPEA2

En las corridas realizadas del algoritmo SPEA2 se utilizaron los siguientes parámetros:

- Tamaño de la población (*nind*) = 100.
- Número máximo de locutorios de cabinas telefónicas (*nmax*) = 30.
- Tamaño del archivo de no dominados (*nptrue*) = 100.
- Número máximo de generaciones (*ngen*) = 300.
- Probabilidad de cruzamiento (*pc*) = 0,7 a 0,9.
- Probabilidad de mutación (*pm*) = 0,1 a 0,3.

Las valoraciones de cada uno de los parámetros citados fueron establecidos considerando los resultados experimentales obtenidos por *Vineet Khare* en su trabajo *Performance Scaling of Multi-Objetive Evolutionary Algorithms* [21], y de acuerdo a las características actuales del mercado de la telefonía pública, en lo que respecta al número mínimo y máximo de locutorios.

A continuación, se presenta el Pseudocódigo del algoritmo Multiobjetivo utilizado:

Programa Principal SPEA2
Leer los parámetros del SPEA2: *nind, nmin, nmax, ngen, pm, pc, nptrue*
Generar una población usando el algoritmo heurístico (Pseudocódigo 1)
Generar un archivo vacío (conjunto externo)
Para *gen*=1 hasta *ngen*
    Eliminar locutorios repetidos del individuo
    Evaluar funciones objetivo de cada individuo de la población
    Asignar *fitness* a cada individuo de la población y del archivo
    Calcular todos los individuos no dominados de la población y el archivo
    Actualizar el archivo con los individuos no dominados
    Si el tamaño del archivo es mayor que *nptrue*
        Reducir el tamaño del archivo con el operador de truncamiento
    Caso contrario
        Si el tamaño del archivo es menor que *nptrue*
            Copiar los mejores individuos dominados del archivo y la población con *fitness* ≥ 1 al archivo de la nueva generación hasta que el tamaño del archivo sea igual a *nptrue*
        Fin Si
    Si *gen* es menor que *ngen*
        Realizar torneo binario para seleccionar los individuos del archivo que formarán parte del conjunto de emparejamientos
        Realizar cruzamiento y mutación del conjunto de emparejamientos
        Actualizar la población del resultado del conjunto de emparejamientos
    Fin Si
    Incrementar contador de generaciones (*gen*=*gen* + 1)
Fin Para
Salvar el archivo (conjunto de no dominados)

Pseudocódigo 2: Algoritmo SPEA2 implementado.

4 Resultados Experimentales

Las soluciones obtenidas para el problema de prueba que son presentadas en esta sección fueron obtenidas mediante sucesivas corridas del algoritmo SPEA2, las cuales tras cada corrida fueron sometidas nuevamente al concepto de dominancia Pareto considerando las soluciones previamente calculadas, para finalmente obtener los resultados presentados en la tabla 2.

Dada la disponibilidad de datos de los locutorios de cabinas telefónicas habilitadas realmente en la ciudad de Asunción [8], se procedió a calcular un conjunto Pareto óptimo de soluciones para cada año, desde el 2001 hasta el 2003. Por razones de espacio solo se presenta el conjunto Pareto correspondiente al año 2002 (ver tabla 2). En las corridas realizadas para cada año, se tuvo en cuenta los locutorios de cabinas telefónicas habilitadas en años anteriores, de manera que al calcular la solución para el año 2003 se tuvieron en cuenta las cabinas realmente habilitadas hasta el año 2002 y así para los años anteriores considerados.

A fin de establecer comparaciones que denoten la validez de los resultados obtenidos con el algoritmo SPEA2, se evaluaron las soluciones efectivamente implementadas en la ciudad de Asunción para cada año del periodo considerado (2001-2003). Estas fueron evaluadas con los mismos criterios que las soluciones Pareto calculadas y son mostradas en la tabla 1.

Tabla 1: Tabla de soluciones efectivamente implementadas en la ciudad de Asunción.

Año	Vector de solución implementado	Costo de inversión (US$)	Ganancias mensuales (US$)	Plazo de recupero (meses)
2001	[77 78 107 170 172 174 179 181 189 198 208 209 231 236 237 244 258 262 265 273 288 292 298 313 321 339 363 364 369 394]	219.840	9.200	36
2002	[139 142 181 182 203 204 206 208 209 221 230 231 234 235 236 237 265 286 287 291 292 320 329 338 340 365 370 373 375 396]	207.890	8.580	36
2003	[0 0 0 0 20 139 140 198 204 220 221 233 242 245 258 289 291 292 298 320 322 339 348 349 354 389 390 391]	150.240	2.610	>120

Dada la evaluación de las soluciones efectivamente implementadas de la tabla 1 y los correspondientes conjuntos Pareto óptimos, se escogió una solución de cada uno de estos conjuntos con montos de inversión semejantes a sus correspondientes soluciones efectivamente implementadas. De este modo se obtuvo un emparejamiento de las soluciones implementadas y las calculadas para cada año, los cuales se detallan en la tabla 3. Se puede observar claramente que conforme fue aumentando el número de locutorios ubicados, las ganancias correspondientes a cada año fue decreciendo (ver tabla 3), hasta el punto que en el año 2003 las estimaciones para recuperar el capital invertido superan ampliamente los 10 años, lo cual constituye un fracaso de la inversión. Del mismo modo, se puede notar que para los dos primeros años considerados, las ganancias estimadas por las soluciones Pareto calculadas son aproximadamente 80% superiores a sus pares implementadas, mientras que los tiempos de recupero de la inversión son reducidos a la mitad del tiempo de aquellas.

Estos datos muestran la alta calidad de las soluciones Pareto calculadas, demostrando que las soluciones efectivamente implementadas constituyen soluciones sub-óptimas para el problema planteado.

Tabla 2: Tabla de soluciones no dominadas encontradas.

	TABLA DE SOLUCIONES NO DOMINADAS 2002																																	OBJETIVOS		
solu.	Vectores de decisiones																																	costo	ganan.	plazo
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	38	109	292	307	338	356	387	466	491	497					52948	1179	90
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	168	186	249	307	387	389	390	468					53154	2281	36
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	41	84	125	164	168	179	205	287	388	497					61950	2284	42
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	99	261	294	307	332	387	390	393	414	468					62522	2450	36
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	20	47	222	338	355	363	366	387	466					63792	2651	36
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	99	141	174	205	253	266	294	364	390	412					65005	2970	30
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	47	109	151	194	287	338	363	366	387	466					67472	3214	30
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	66	121	204	205	253	266	294	338	390	469					68197	3529	30
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	66	121	204	205	224	253	266	294	338	390					70660	3614	30
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	47	109	156	287	338	363	366	390	444	466					72186	3747	30
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	41	79	142	143	266	287	364	388	390	412					73872	3880	24
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	186	222	266	342	363	364	387	390	412	430					74580	4059	24
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	109	151	204	287	338	350	354	363	366	387					77060	4310	24
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	67	188	204	266	295	338	362	366	390	417					77847	4830	24
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	122	204	266	287	308	338	362	364	390	442					80210	5317	24
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	22	151	204	266	287	338	354	364	366	387	390					88502	5482	24
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	122	204	232	266	287	308	338	362	364	442	487					88556	5671	24
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	67	122	151	204	266	287	338	362	364	366	390					90157	5849	24
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	151	204	266	287	338	354	363	364	366	387	390					90902	6042	24
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	188	204	205	266	287	338	364	366	390	412	475					93352	6125	18
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	109	205	287	318	338	362	364	366	389	390	444	451					96883	6165	24
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	151	204	266	287	318	338	354	363	364	366	387	390					97659	6354	24
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	13	186	188	205	266	287	338	362	364	366	390	412					99286	6441	18
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	63	205	266	278	287	338	341	362	363	364	366	444					100059	6501	24
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	188	204	205	266	284	287	338	364	366	390	412	475					100884	6556	24
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	109	205	266	287	317	337	362	363	364	366	390	444					102518	6779	18
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	188	204	205	266	287	338	362	364	366	390	412	496					103272	6897	18
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	164	204	244	266	286	287	338	341	364	366	389	390					105043	7210	18
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	205	266	287	317	337	338	362	363	364	366	390	444					105632	7386	18
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	71	203	204	232	266	286	338	341	351	364	366	389	390					110017	7455	18
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	178	204	266	294	307	338	341	342	362	364	366	389	390					111582	7734	18
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	62	204	205	266	287	338	341	362	366	389	412	444	461					117177	7835	18
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	147	205	259	266	287	317	337	338	362	363	364	390	415	444				118319	7900	18
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	114	263	266	287	317	338	341	362	364	366	388	389	390	451				121536	8050	18
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	178	204	266	293	307	321	338	341	342	362	364	366	389	390	420				123486	8106	18
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	204	205	209	232	266	287	338	341	362	364	366	372	383	389	497				124354	8122	18
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	122	204	232	257	266	287	299	362	364	366	386	387	388	390	421				124383	8224	18
38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	152	204	206	233	266	287	317	341	343	364	366	387	389	390				126409	8366	24
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	98	205	232	263	266	287	317	326	337	338	362	364	388	390	444				128317	8423	18
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	114	263	266	287	317	338	341	362	364	366	388	389	390	444	451				129061	8439	18
41	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	20	70	203	205	232	257	266	287	337	338	362	366	372	388	389				129837	8705	18
42	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	188	204	205	232	263	266	287	337	338	364	366	388	390	412	475				133110	8996	18
43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	205	232	263	266	287	317	326	337	338	362	364	366	388	390	444				134750	9170	18
44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	20	205	257	266	287	337	338	362	364	365	366	372	388	389	390	421				138144	9438	18
45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	20	205	232	257	266	287	337	338	362	364	366	372	388	389	390	421				141352	9757	18
46	0	0	0	0	0	0	0	0	0	0	0	0	0	186	232	257	266	287	320	337	338	362	364	366	372	388	389	390	421					142196	9912	18
47	0	0	0	0	0	0	0	0	0	0	0	0	0	159	204	205	232	257	266	287	338	341	362	364	366	372	383	388	389	390				146523	10181	18
48	0	0	0	0	0	0	0	0	0	0	0	0	134	180	232	260	266	287	313	316	317	338	362	363	364	366	388	390	437	444				154920	10428	18
49	0	0	0	0	0	0	0	0	0	0	0	0	147	204	260	263	266	284	287	302	307	315	338	362	364	366	388	389	390	425				158125	10468	18
50	0	0	0	0	0	0	0	0	0	0	0	0	61	204	260	263	266	284	287	302	307	315	338	341	362	364	366	388	389	390				159751	10651	18
51	0	0	0	0	0	0	0	0	0	0	0	136	178	204	266	278	286	287	307	316	337	338	341	342	362	364	366	375	388	389				163224	10673	18
52	0	0	0	0	0	0	0	0	0	0	0	136	178	204	223	232	266	286	287	307	337	338	341	342	362	364	366	377	388	389				164841	10934	18
53	0	0	0	0	0	0	0	0	0	0	0	17	114	232	263	266	286	287	338	341	342	362	364	366	388	389	390	414	444	461				168812	11098	18
54	0	0	0	0	0	0	0	0	0	0	43	178	204	232	259	266	286	287	308	337	338	341	362	364	366	388	389	390	414	480				169088	11520	18
55	0	0	0	0	0	0	0	0	0	72	107	178	204	232	259	266	286	337	338	341	342	362	364	365	366	388	389	390	414	425				178276	12242	18
56	0	0	0	0	0	0	0	0	43	162	178	204	232	259	266	286	287	308	314	337	338	341	362	364	365	366	388	389	390	480				185847	12717	18
57	0	0	0	0	0	0	43	162	204	232	257	259	266	286	287	290	308	314	337	338	341	362	364	365	366	371	389	390	430	480				199377	12932	18
58	0	0	0	0	0	0	43	178	204	232	238	259	266	286	287	307	333	337	338	341	342	362	364	366	388	389	390	413	414	461				207267	13592	18
59	0	0	0	0	43	178	204	207	232	238	259	266	284	286	287	307	333	337	338	341	342	362	366	388	389	390	397	413	414	461				219202	13836	24

Tabla 3: Tabla de emparejamiento de las soluciones efectivamente implementadas y las calculadas.

Año	Soluciones Consideradas	Costo Inversión (US$)	Ganancias mensuales (US$)	Plazo de Recupero (meses)	Locutorios Ubicados
2001	Solución implementada	219.840	9.200	36	30
	Solución calculada con SPEA2	216.326	15.014	18	24
2002	Solución implementada	207.890	8.580	36	30
	Solución calculada con SPEA2	199.377	12.932	18	24
2003	Solución implementada	150.240	2.610	> 120	26
	Solución calculada con SPEA2	145.523	9.812	18	16

En el conjunto Pareto calculado para cada uno de los años considerados, se pueden encontrar otras soluciones que con inversiones muy inferiores a lo efectivamente implementado se pueden obtener ganancias muy superiores a las que se obtuvieron en la realidad, como se puede notar de la tabla 1 y tabla 2, para el año 2002.

Los valores de las matrices, datos y diagramas utilizados en los resultados experimentales presentados están disponibles en [2].

5 Conclusiones

La utilización de Algoritmos Evolutivos Multiobjetivos en la resolución de problemas de ubicación de centros proveedores de servicios, como locutorios de cabinas telefónicas para este caso particular, presenta un nuevo enfoque en la solución de los llamados *facility location problems*. Esta metodología proporciona una herramienta computacional que permite obtener un conjunto de soluciones Pareto óptimas, considerando simultáneamente todos los aspectos que se quieran optimizar. Los métodos heurísticos tradicionales [17] o de algoritmos de aproximación [9] proporcionan soluciones mono-objetivos, restringiendo de este modo el espectro del planificador al momento de tomar una decisión.

Claro está que en la ubicación de los centros proveedores de servicios, intervienen cada vez más factores que necesitan ser tenidos en cuenta para lograr la mejor eficiencia, no solamente en la reducción de los costos que implican su habilitación, sino tambien en otros aspectos que hacen a la planificación de una buena distribución de recursos, con lo cual se abre paso a la utilización de los Algoritmos Evolutivos Multiobjetivos como el SPEA2. Conforme a los resultados obtenidos en este trabajo, se puede afirmar que las soluciones del frente Pareto correspondiente a cada año considerado para la ubicación de los locutorios de cabinas telefónicas son dominantes con respecto a las soluciones implementadas en la realidad, y de hecho, en las pruebas realizadas, las soluciones obtenidas demuestran claramente que las inversiones practicadas en la habilitación de estos locutorios para la ciudad de Asunción hubiesen dado mejores dividendos con la utilización de esta herramienta. Cabe mencionar que en el periodo considerado, el crecimiento de la oferta de servicios de telefonía pública se dio de manera acelerada y sin la orientación de una herramienta como la que propone el presente trabajo, con la consecuente ubicación de locutorios de cabinas telefónicas en zonas donde la oferta supera a la demanda y en cotrapartida dejando sin servicio a otras zonas donde la demanda realmente amerita la ubicación de esos locutorios. Como lógica consecuencia, en la actualidad se van cerrando muchos de los locutorios debido a la baja rentabilidad de los mismos por el criterio intuitivo a que obedecieron su ubicación, mientras nuevas cabinas se abren en zonas con demanda no atendida.

En definitiva, se puede aseverar que el empleo de algoritmos evolutivos multiobjetivos para el tipo de problema considerado representa una herramienta muy eficiente, que permite a los planificadores manejar diversos aspectos del problema para la optimización de varios objetivos, además de proveer un panorama más amplio al momento de la toma de decisiones, teniendo en cuenta que se calcula un conjunto de soluciones óptimas de compromiso y no una solución única.

Cabe destacar que la metodología adoptada para resolver el problema de ubicación de locutorios de cabinas telefónicas es fácilmente adaptable a otros problemas similares, además de la ubicación de centrales telefónicas [1]. Por ejemplo: ubicación de estaciones radio bases para telefonía celular, o en general, ubicación de manera óptima de centros de atendimiento de diversos servicios, como cadenas de comida rápidas, supermercados, etc. La simplicidad de la metodología propuesta, para un problema tan complejo, alienta a mirar con optimismo la realización de futuros trabajos en el área, así como nuevas aplicaciones.

Referencias

[1] Almeida C., Amarilla N. y Barán B.: Optimización Multiobjetivo en la Planificación de Centrales Telefónicas. *XXIX Conferencia Latinoamericana de Informática CLEI2003*, La Paz, Bolivia, 2003.

[2] Amarilla N., Almeida C. y Barán B.: Reporte Técnico *01/2004. Centro Nacional de Computación, Universidad Nacional de Asunción*. San Lorenzo, Paraguay. Enero, 2004.

[3] Arroyo J. y Armentano V.: Um Algoritmo Genético para Problemas de Otimizaça• o Combinatória Multiobjetivo, *XXXIII Simpósio Brasileiro de Pesquisa Operacional*. Campos do Jordao – SP. Noviembre, 2001.

[4] A. S. Manne: Plant location under economies of scale decentralization and computation. *Management Science*, Vol. 11, pag. 213-235, 1964.

[5] Barán B. y Duarte S.: Multiobjective Network Design Optimization using Parallel Evolutionary Algorithms. *XXVII Conferencia Latinoamericana de Informática CLEI2001*. Mérida - Venezuela. 2001.

[6] Bumb A.: Approximation algorithms for facility location problems. PhD thesis, *University of Twente*, Netherland, Octubre 2002.

[7] Boorstyn R. R. y Frank H.: Large-Scale Network Topological Optimization. *IEEE Transactions on Communications*, Vol. 25, pag. 29-47, Enero 1977.

[8] Comisión Nacional de Telecomunicaciones CONATEL: http://www.conatel.gov.py.

[9] David B., Eva Tardos y Karen Aardal: Approximation algorithms for facility location problems (extended abstract). In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pag. 265-274, El Paso, Texas, Mayo 1997.

[10] Dirección General de Estadísticas: Encuestas y Censos: *Sistema Estadístico Nacional*, CD de población y viviendas. Paraguay. 1997.

[11] Feldman E., Lehner F. A., y Ray T. L.: Warehouse Locations Under Continuous Economies of Scale, *Management Science*, Vol. 12, pag. 670-684, Mayo 1966.

[12] Glover F., Laguna M., Taillard E., y Werra D. De: Tabu Search, special issues of Annals of Operations Research, Vol. 41, J. C. *Baltzer Science Publishers*, Basel, Switzerland, 1993.

[13] Joseph A. Pecar, David A. Garbin. Te New Telecom Factbook. *The Mc Graw-Hill*, segunda edición, pag. 415, Copyright 2000.

[14] J. Bar-IIan, G. Kortsarz y Peleg.: How to allocate network centers. *Journal of Algorithms*. Vol. 15, pag. 385-415, noviembre 1993.

[15] J. F. Stollsteimer: The effect of technical change and output expansion on the optimum number, size and location of pear marketing facilities in a California pear producing region. PhD thesis, *University of California at Berkeley*, Berkeley, California, 1961.

[16] J. F. Stollsteimer: A working model for plant numbers and locations. *Journal of Farm Economics,* Vol. 45, pag. 631-645, 1963.

[17] Kuehn A. A. y Hamburger M.J.: "A Heuristic Program for Locating Warehouses", *Management Science,* Vol. 9, pag. 643-666, 1963.

[18] M. L. Balinsksi: On finding integer solutions to linear programs. *In Proceedings of the IBM Scientific Computing Symposium on Combinatorial Problems*, pages 225-248. IBM, 1966.

[19] Nippon Telephone and Telegraph: "Planning Telecommunication Network", *Japan International Cooperation Agency*, Japan,1998.

[20] Robert C. Merton. FINANZAS: *Prentice Hall*, Mexico, mayo 1999

[21] Vineet Khare: "Performance Scaling of Multi-Objective Evolutionary Algortihms", *School of Computer Science, University of Birmingham*. Edgbaston, Birmingham B15 2TT, U.K., setiembre 2002.

[22] Zitzler E., Laumanns M., y Thiele L.: SPEA 2: Improving The Strength Pareto Evolutionary Algorithms, *Technical Report 103*, *Computer Engineering and Networks Laboratory*, *Swiss Federal Institute of Technology*. Zurich, Switzerland, Mayo 2001.

ISAM: Uma Arquitetura de Software para *Pervasive Computing*

Adenauer C. Yamin[1], Jorge L. V. Barbosa[1,2], Iara Augustin[3], Luciano C. da Silva[4], Rodrigo A. Real[4], Cláudio F. R. Geyer[4]

[1]*Escola de Informática*
Universidade Católica de Pelotas
Pelotas, RS, Brasil
{adenauer,barbosa}@ucpel.tche.br

[2]*Engenharia da Computação*
Universidade do Vale do Rio dos Sinos
São Leopoldo, RS, Brasil
barbosa@exatas.unisinos.br

[3]*Departamento de Eletrônica e Computação*
Universidade Federal de Santa Maria
Santa Maria, RS, Brasil
august@inf.ufsm.br

[4]*Instituto de Informática*
Universidade Federal do Rio Grande do Sul
Porto Alegre, RS, Brasil
{lucc,rreal,geyer}@inf.ufrgs.br

Abstract

The next years will be characterized for high levels of mobility, heterogeneity and interactions among devices connected to global networks. These interconnected networks will use as much wired connections as wireless. The first researches involving wide-area distributed systems answered many questions concerning resource management, although they fail in treating questions related to heterogeneity and dynamic adaptation. More recent works, through technologies like CORBA and Java/Jini, deal with heterogeneity problem, but do not deepen into adaptability aspects. In this article, we propose inside of the project ISAM, a treatment of this subject. ISAM contemplates an integrated approach of software and execution environment addressed to the administration of adaptation in heterogeneous networks, supporting the logical (software) and physical (hardware) mobility, oriented to the execution of distributed applications in a global scale and based on software components.

Keywords: Network Computers, Distributed Systems.

Resumo

Os próximos anos serão caracterizados por elevados níveis de mobilidade, heterogeneidade e interação entre dispositivos conectados a redes globais. As primeiras pesquisas envolvendo sistemas distribuídos em redes wide-area, responderam a diversas questões pertinentes ao gerenciamento de recursos. Trabalhos mais recentes empregando tecnologias como CORBA e Java/Jini abordam a questão da heterogeneidade, porém não se aprofundam em aspectos pertinentes à adaptabilidade. Neste artigo, apresenta-se o modelo ISAM, uma proposta para adaptação em redes heterogêneas, com suporte às mobilidades de software e hardware, voltada à execução de aplicações distribuídas em escala global e baseada em componentes de software.

Palavras-chave: Redes de Computadores, Sistemas Distribuídos.

1 Introdução

Nos últimos anos, os avanços da microeletrônica vêm diminuindo o preço do hardware e aumentando seu poder computacional. Além disso, o desenvolvimento de soluções eficientes para interconexão dos sistemas computacionais fez com que a área de redes de computadores assumisse uma posição de destaque. O crescimento exponencial da *Internet* vem sendo considerado um fenômeno tecnológico e de mercado. Neste contexto, as plataformas computacionais vêm migrando de sua natureza centralizada para uma nova realidade distribuída. Esta nova perspectiva de processamento em rede assume diferentes perspectivas: *Metacomputing*, *Grid Computing*, *Internet Computing* e mais recentemente *Peer-to-Peer Computing* [18].

Atualmente, os estudos sobre mobilidade em sistemas distribuídos são impulsionados pela proliferação de dispositivos eletrônicos portáteis. Este novo paradigma computacional distribuído e móvel é denominado computação móvel. Amplia-se assim o conceito de rede-sem-fio. Nesta nova perspectiva, o usuário portando dispositivos móveis como *palmtops* e *notebooks*, independentemente da sua localização física, terá acesso a uma infra-estrutura de serviços [4].

Neste contexto observa-se um movimento em direção à *Pervasive Computing* [22], onde novas aplicações atendem as necessidades dos usuários que se deslocam. *Pervasive Computing* é a proposta de um novo paradigma computacional, que permite ao usuário o acesso ao seu ambiente computacional a partir de qualquer lugar, a qualquer tempo, usando vários tipos de dispositivos (móveis ou não). Nesse sentido, a aplicação ou o ambiente de execução pró-ativamente monitoram e controlam as condições do contexto. A aplicação reage às alterações no contexto através do processo de adaptação. Este processo requer a existência de múltiplos caminhos de execução para uma mesma aplicação ou configurações alternativas que exibam diferentes perfis de utilização (históricos) dos elementos computacionais.

Esta visão apresenta uma série de novos (e renovados) desafios, oriundos do dinamismo e heterogeneidade do ambiente, além dos novos requerimentos das aplicações no estilo *follow-me* da *Pervasive Computing*. O dinamismo está tanto na aplicação quanto no sistema de execução. Ambos operam em um ambiente cujas condições na disponibilidade e no acesso aos recursos são variáveis no tempo e no espaço. Como conseqüência, novos tipos de aplicações estão aparecendo, as quais têm um comportamento determinado pela sua sensibilidade à variação nas condições de alguns elementos do ambiente. O ambiente é definido por elementos computacionais que podem ser medidos, como largura de banda, latência da rede, consumo de energia, localização do usuário, preferências do usuário, entre outros [10]. A complexidade do tratamento da adaptação introduz custos, tanto no gerenciamento, como na execução da aplicação. Estes custos podem se tornar elevados devido à necessidade de predições e estimativas de comportamentos futuros. Por outro lado, os ganhos potenciais podem ser altos. Davies [24] acredita que somente através de um processo de adaptação baseado em informações gerenciais sobre trocas na infra-estrutura de suporte, será possível operar eficientemente em um ambiente distribuído altamente dinâmico.

É no cenário da *Pervasive Computing* que a necessidade da adaptação se potencializa. Somente através de um mecanismo eficiente de adaptação, as aplicações executando em um computador móvel podem gerenciar questões tais como: variação imprevisível na qualidade da rede, grande disparidade na disponibilidade de serviços remotos e limitações nos recursos locais impostos pelas restrições de peso, pelo tamanho dos dispositivos móveis e pelo consumo de bateria.

Sistemas distribuídos tradicionais são construídos com suposições sobre a infra-estrutura física de execução, como conectividade permanente e disponibilidade dos recursos necessários. Porém, essas suposições não são válidas na *Pervasive Computing* [22]. Conforme o usuário se movimenta, a localização do seu dispositivo móvel se altera, e conseqüentemente, a configuração da rede de acesso e o centro da atividade computacional também se modificam. A mobilidade de hardware impede o uso direto das soluções adotadas pelos sistemas distribuídos atuais, construídos com premissas que envolvem somente a mobilidade de software.

Neste novo cenário, o comportamento adaptativo das aplicações levanta um conjunto de questões: (*i*) "quais são os domínios de aplicações adequados?"; (*ii*) "como os programas devem ser estruturados?"; (*iii*) "como o sistema básico de suporte deve trabalhar ?". O modelo ISAM (Infraestrutura de Suporte às Aplicações Móveis Distribuídas) aborda essas questões. O ISAM contempla uma arquitetura de software que enfoca de forma integrada os problemas da adaptação considerando cenários onde as tarefas da aplicação apresentam a propriedade de mobilidade. Esta mobilidade pode ser tanto física (equipamento se desloca) como lógica (o software transita entre os equipamentos).

Este artigo aborda particularmente uma das questões da arquitetura ISAM: a adaptação colaborativa entre a aplicação (programa) e o ambiente de execução (*middleware*). Na arquitetura ISAM, o sistema se adapta para fornecer qualidade, enquanto que a aplicação se adapta para manter a qualidade dentro da expectativa do usuário móvel. A estrutura do texto é a seguinte. Na seção 2 é apresentada a arquitetura de software ISAM. A seção 3 apresenta o modelo de adaptação proposto. Na seção 4 é discutida uma aplicação. A seção 5 analisa os trabalhos relacionados, e as considerações finais são apresentadas na seção 6.

2 Arquitetura de Software ISAM

A arquitetura proposta é organizada em camadas com níveis diferenciados de abstração. O ISAM busca a manutenibilidade da qualidade de serviços oferecida ao usuário móvel, através do conceito de adaptação. Uma visão organizacional desta arquitetura é apresentada na figura 1. Salientam-se dois pontos: (*i*) a adaptação que permeia todo o sistema, por isto está colocada em destaque; (*ii*) o escalonador, que é parte central do EXEHDA (*middleware* ISAM) [3].

A camada superior (SUP) da arquitetura é composta pela aplicação móvel distribuída. A construção desta aplicação baseia-se nas abstrações do Holoparadigma [15] [16], as quais permitem expressar mobilidade, acrescidas de novas abstrações para expressar adaptabilidade providas pelo ISAMadapt [9].

Por sua vez, a camada inferior (INF) é composta pelas tecnologias empregadas nos sistemas distribuídos existentes, tais como sistemas operacionais nativos e a Máquina Virtual Java.

2.1 Camada Intermediária – Primeiro Nível

A camada intermediária (INTERM) é o núcleo funcional da arquitetura ISAM, sendo formada por três níveis de abstração. O primeiro nível é composto por dois módulos de serviço à aplicação: Escalonamento e Ambiente Virtual do Usuário. O escalonamento, por sua vez, é o componente-chave da adaptação na arquitetura ISAM.

O Ambiente Virtual do Usuário (AVU) compõe-se dos elementos que integram a interface de interação do usuário móvel com o sistema. Este módulo permite que uma aplicação sendo executada em uma localização possa ser instanciada e continuada em outra localização sem interrupção, suportando assim o estilo de aplicações *follow-me*. O modelo foi projetado para suportar a exploração de aplicações contextualizadas (adaptadas aos recursos, serviços e localização corrente) e individualizadas (adaptadas aos interesses e preferências do usuário móvel). O desafio da adaptabilidade é suportar os usuários em diferentes localizações, com diferentes sistemas de interação que demandam diferentes sistemas de apresentação, dentro dos limites da mobilidade. Este módulo deve caracterizar, selecionar e apresentar as informações de acordo com as necessidades e o contexto em que o usuário se encontra. Para realizar estas tarefas, o sistema se baseia num modelo de uso onde as informações sobre o ambiente de trabalho, preferências, padrões de uso, padrões de movimento físico e hardware do usuário são dinamicamente monitoradas e integram o Perfil do Usuário e da Aplicação.

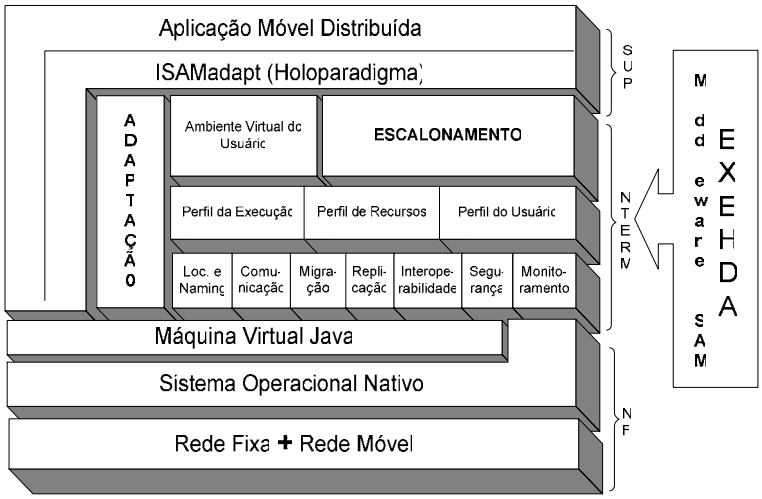


Figura 1. Arquitetura ISAM

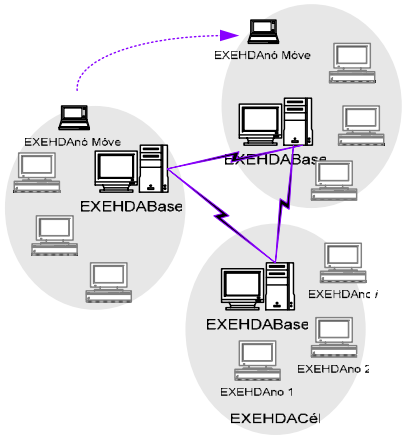


Figura 2. Ambiente de Execução ISAM

2.2 Camada Intermediária – Segundo Nível

Como já caracterizado anteriormente, na proposta ISAM busca-se um conceito flexível de adaptação que está relacionado ao contexto em que a aplicação está inserida. Por sua vez, a mobilidade de hardware introduz a possibilidade de movimentação do usuário durante a execução de uma aplicação. Desta forma, os recursos

disponíveis podem se alterar, tanto em função da área de cobertura e heterogeneidade das redes, quanto em função da disponibilidade dos recursos devido à alta dinamicidade do sistema. Assim, a localização corrente do usuário determina o contexto de execução, definido como "toda informação, relevante para a aplicação, que pode ser obtida e usada para definir seu comportamento" [9]. Numa análise preliminar, o contexto é determinado através de informações de quem, onde, quando, o que está sendo realizado e com o que está sendo realizado. Obter essas informações é a tarefa do módulo de monitoramento, que atua tanto na parte móvel quanto na parte fixa da rede.

As informações que dirigem as decisões do escalonador e dão suporte à aplicação para sua decisão de adaptação são advindas de quatro fontes: perfil da execução, perfil dos recursos, perfil do usuário e da aplicação (ISAMadapt). O módulo monitoramento do ISAM obtém informações do acompanhamento das aplicações executadas pelo usuário, em um dado tempo e em um dado local, com determinados parâmetros, o que permite determinar a evolução histórica e quantitativa das entidades monitoradas. A interpretação destas informações estabelece o perfil do usuário e das aplicações. Desta forma, as aplicações móveis ISAM poderão se adaptar à dimensão pessoal, além das dimensões temporal e espacial presentes nos demais sistemas móveis [10].

2.3 Camada Intermediária – Terceiro Nível

No terceiro nível da camada intermediária estão os serviços básicos do ambiente de execução ISAM que provêm a funcionalidade necessária para o segundo nível e cobrem vários aspectos, tais como migração – mecanismos para deslocar um ente de uma localização física para outra; replicação otimista – mecanismo para aumentar a disponibilidade e o desempenho do acesso aos dados; localização e *naming* – para dar suporte ao movimento dos dispositivos móveis entre diferentes *EXEHDACéls* (veja figura 2), mantendo a execução durante o deslocamento.

3 A Adaptação Multinível Colaborativa no ISAM

A proposta ISAM contempla um comportamento adaptativo em dois segmentos: (*i*) na aplicação, a qual define o comportamento da adaptação (alternativas) e o contexto de seu interesse; (*ii*) no ambiente de execução (*middleware*), o qual tem um comportamento inerentemente adaptativo no momento em que processa a aplicação. Uma visão estrutural da Adaptação Multinível Colaborativa proposta no ISAM pode ser vista na figura 3.

Na codificação da aplicação, o programador especifica os elementos computacionais que afetam o comportamento da aplicação, os respectivos níveis de variação suportados e codifica comportamentos alternativos para atender à variação nas condições ambientais (variação nos elementos computacionais no contexto da aplicação).

Estas informações influenciam o comportamento adaptativo a ser adotado pelo ambiente de execução. Por sua vez, o ambiente de execução (EXEHDA - *middleware* ISAM) fornece meios: (*a*) para que sejam monitorados elementos computacionais do ambiente, (*b*) para que a aplicação possa registrar seu interesse em determinados elementos, (*c*) para notificar à aplicação das alterações ocorridas, e (*d*) para selecionar o comportamento alternativo mais adequado ao ajuste das novas condições ambientais. Como o sistema gerencia as aplicações, este também pode ter um comportamento pró-ativo,e executar adaptações relativas à administração e desempenho do sistema de forma global.

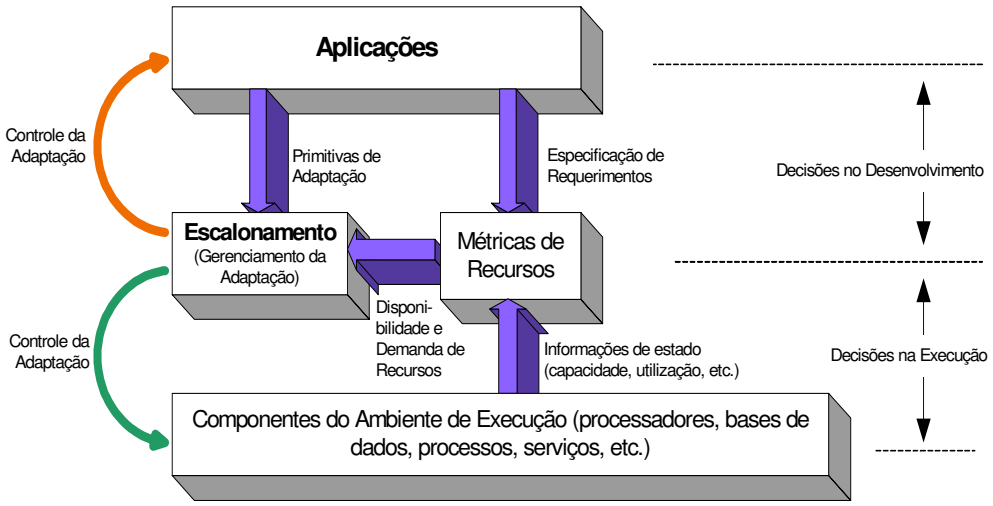


Figura 3. A Adaptação Multinível Colaborativa no ISAM

3.1 Adaptação no Nível da Aplicação

No ISAM o desenvolvimento de aplicações emprega as abstrações do Holoparadigma e do ISAMadapt através da HoloLinguagem para expressar o comportamento adaptativo da aplicação. A Hololinguagem [16] é uma linguagem de programação que integra os paradigmas em lógica, imperativo e orientado a objetos. A mesma utiliza um modelo de coordenação que suporta invocações implícitas (*blackboard*). Este modelo de coordenação torna a linguagem apropriada ao ISAM, pois um requisito da *Pervasive Computing* é o desacoplamento temporal e espacial da comunicação [9]. A natureza da mobilidade não garante a interação contínua entre os componentes de uma aplicação distribuída neste ambiente. Desconexões são comuns, não somente devido ao meio físico mas sobretudo como uma estratégia para economia de energia nos dispositivos móveis. Logo, o ambiente móvel requer mecanismos de coordenação sem a premissa da conexão permanente. Além disso, devido à natureza imprecisa da mobilidade, o modelo de coordenação deve enfatizar a comunicação assíncrona e anônima.

A Hololinguagem suporta ainda concorrência, modularidade, mobilidade e encapsulamento de *blackboards* em tipos abstratos de dados. No Holoparadigma [15], a aplicação é modelada com entes (entidade de existência) e símbolos (entidade de informação). Existem dois tipos de entes: elementar e composto. Um ente elementar é organizado em três partes: *Interface*, *Comportamento* e *História*. A interface descreve suas possíveis relações com os demais entes. O comportamento contém ações que implementam sua funcionalidade. Por sua vez, a história é um espaço de armazenamento compartilhado no interior de um ente. Um ente composto possui a mesma organização do ente elementar, no entanto, suporta a co-existência de outros entes na sua composição (entes componentes).

A Adaptação Multinível Colaborativa no nível de aplicação é implementada através das abstrações expressas na HoloLinguagem e da utilização das informações fornecidas pelo Monitoramento do Contexto (MC). Destaca-se aqui a existência dos *Holosensores* (seção 0), responsáveis por capturar informações que alimentam a tomada de decisões na arquitetura. Os *Holosensores* atuam em duas instâncias:

> De sistema. Têm-se dois tipos principais de métricas: (*i*) uma para caracterização do *workload* dos *hosts* e (*ii*) outra para construção de perfis de comunicação entre objetos. Os mecanismos para capturar os perfis de comunicação entre os objetos são integrados com a API RMI de Java, preservando a compatibilidade com a semântica nativa da linguagem [17];
>
> De aplicação. É feita uma monitoração utilizando a JVMPI (*Java Virtual Machine Profiler Interface*), a qual oferece a possibilidade de uma seleção dinâmica dos eventos de interesse da aplicação (por exemplo, ativação, interrupção e tempo de espera de métodos) que devem ser monitorados. No ISAM, esta monitoração não exige cuidados de programação, e os eventos a serem monitorados podem ser individualmente ativados e/ou desativados para redução de *overheads* desnecessários [6].

O MC é definido para manipular todos os recursos de interesse do sistema e das aplicações que podem afetar a arquitetura. Além disso, como as aplicações têm interesse em contextos particulares e podem requerer diferentes interpretações para os mesmos dados capturados (individualização do contexto), fez-se necessário definir uma abstração que definisse para a arquitetura os elementos do contexto de interesse específico das diversas aplicações: *HoloContexto da Aplicação* (HCA).

Para atender o requisito de independência, o sistema de monitoramento de recursos é desacoplado da aplicação, o que induz à necessidade de que as informações que este origina devam ser persistentes. Além disso, uma parte importante do contexto é a informação histórica que pode ser usada para deduzir o comportamento futuro. Esta predição é encapsulada na abstração *Interpretador do HoloContexto* (IHC), o qual produz uma informação parametrizada a partir de uma ou mais informações de contexto. Os geradores das informações de contexto são os *HoloSensores* (Hsensor), que são organizados de forma distribuída, próximos à fonte física/lógica que monitoram, embora sejam vistos como uma unidade pela aplicação (transparência). Os sensores devem trabalhar independentemente da aplicação, pois freqüentemente fornecerão informações para múltiplas aplicações que estão em execução.

O uso das informações de contexto requer abstrações adicionais, pois estas informações freqüentemente não estão na forma requerida pela aplicação. Por exemplo, um sistema GPS (*Global Positioning System*) fornece a informação na forma latitude-longitude; considerando a aplicação, esta pode requerer o nome da rua onde o usuário está. As aplicações trabalham em termos de informações de contexto em alto nível, com mnemônicos como "trabalho", "casa", "rede rápida", "carga baixa", que são fornecidas pelo MC, através da operação de interpretação do IHC. O modelo de contexto definido pelo projetista da aplicação (integrante do ISAMadapt) insere as regras de tradução das informações dos sensores em informações de alto nível consumidas pela aplicação. No exemplo, uma regra poderia ser posição GPS -> CEP -> <casa, rua, escritório>. O comportamento adaptativo da aplicação ISAM está esquematizado na figura 4.

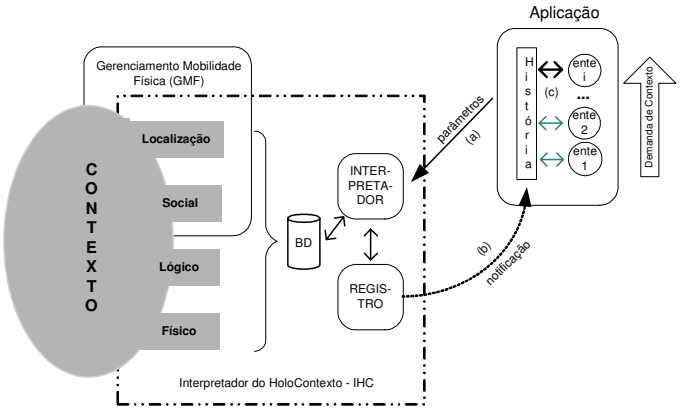


Figura 4. Adaptação no Nível de Aplicação

A aplicação é modelada com a adaptação em mente, onde o projetista implementa entes com funcionalidades semelhantes, porém com demandas de contexto diferentes. As aplicações, através do processo de REGISTRO do IHC, indicam os elementos do contexto que são de seu interesse (*ação a*). Junto com essa requisição de registro, a aplicação passa parâmetros utilizados pelo IHC para fazer a interpretação e notificação, quando houver uma alteração no contexto monitorado (*ação b*). O IHC formata as informações valoradas do contexto para a forma parametrizada pela aplicação. Por exemplo, o nível de bateria pode ser formatado como "alto", "médio", "baixo". O sistema de notificação escreve no *blackboard* do ente que solicitou o registro a informação [bateria, baixa]. O mecanismo de invocação implícita do Holoparadigma [15] é acionado por esta escrita, e o ente que esperava pela notificação é ativado (*ação c*). Desta forma, o controle da adaptação é feito de forma automática pela invocação implícita do ente adequado ao contexto em questão.

3.2 Adaptação no Nível do Sistema

As aplicações ISAM solicitam, direta ou indiretamente, recursos do *middleware ISAM.* Algumas podem especificar uma determinada necessidade de qualidade de serviço (QoS), outras podem aceitar o "melhor-possível" nos níveis de serviço [19]. O escalonador é o artífice central no gerenciamento dos recursos e serviços, e a discussão sobre a estratégia de adaptação será feita com foco no mesmo.

A arquitetura ISAM gerencia diversas aplicações que concorrem na utilização dos recursos. A otimização da execução de um subconjunto do total de aplicações não pode comprometer o nível mínimo de QoS necessário para o restante. Desta forma, o escalonador precisa trabalhar com uma visão global das execuções em andamento. Neste caso, as estratégias de adaptação exigem do mecanismo de escalonamento o tratamento de problemas de otimização utilizando critérios múltiplos [3].

3.2.1 Organização Física do Middleware ISAM

A forma como é organizada a distribuição lógica dos equipamentos afeta diretamente todos os serviços do *middleware*, e naturalmente o escalonamento. O ISAM utiliza uma organização fisicamente distribuída e cooperativa, representada na figura 5. A proposta está baseada em dois escalonadores: (*i*) **EscCél** e (*ii*) **EscEnte**:

EscCél: fica localizado no nodo *EXEHDAbase* e atua entre as *EXEHDACéls*. Suporta atribuições no gerenciamento global da arquitetura, tais como:

- localizar recursos (hardware e software) mais próximos, para reduzir custos de comunicação;
- decidir quando e onde replicar serviços e/ou componentes de software (entes);
- decidir quando e para onde migrar os componentes de software;
- instanciar o Ambiente Virtual do Usuário nos *EXEHDAnodos*. Esta instanciação é feita sob duas óticas: (*i*) balanceamento de carga - neste caso é escolhido o nodo menos carregado, (*ii*) aspectos de afinidade da aplicação - exigência de memória, bases de dados, etc.;
- disponibilizar antecipadamente, por usuário, a demanda de componentes das aplicações e dos dados;
- repassar ao escalonador EscEnte a carga de trabalho (componentes de software) proveniente de outras *EXEHDABases*.

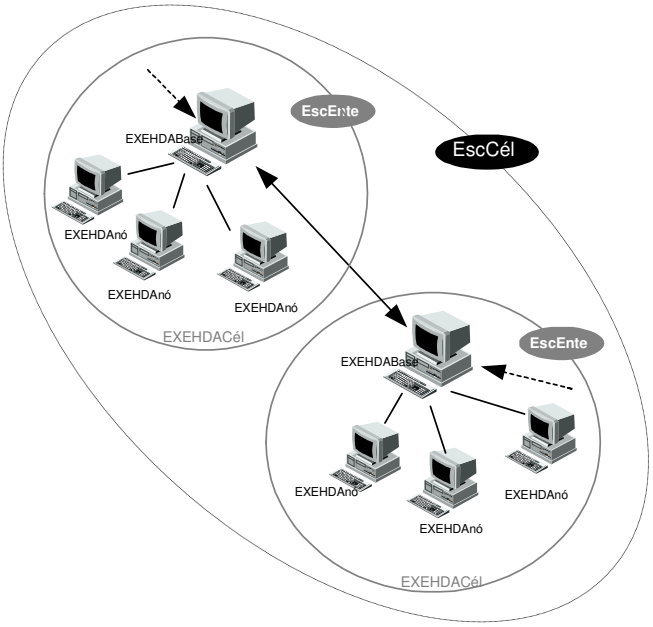


Figura 5. Organização do Escalonamento no Middleware ISAM

Pelas suas atribuições, além da consideração de custos de comunicação e balanceamento de carga, o escalonador EscCél atua de forma intensiva sobre aspectos de replicação e migração.

EscEnte: também existente em todas as *EXEHDABases*, tem atribuições no gerenciamento interno da *EXEHDACél*, tais como:

- efetuar o mapeamento dos componentes da aplicação nos *EXEHDAnodos* da *EXEHDACél*. Os critérios utilizados são balanceamento de carga e afinidade funcional;
- dar suporte aos procedimentos de adaptação colaborativa multinível com a aplicação.

Uma estratégia do escalonador EscEnte é associar o contexto (a *EXEHDACél*, as aplicações e os usuários) a grupos de escalonamento, onde cada grupo pode definir políticas especificas de balanceamento de carga. Cabe ao mecanismo de escalonamento gerir a evolução da execução das aplicações dos diferentes grupos segundo as políticas por ele selecionadas [8].

3.2.2 Principais Macro-Heurísticas Utilizadas

No ISAM, a adaptação permeia todas as decisões da arquitetura. O escalonador é tanto adaptador quanto adaptativo, ou seja, o escalonador é responsável pela execução do comportamento adaptativo da arquitetura, e ele próprio se adapta – altera-se conforme o ambiente corrente. As macro-heurísticas utilizadas pelo escalonador para adaptar-se são:

Aprendizado por reforço: à medida que o usuário interage com o sistema, seu comportamento é monitorado e seu perfil é construído. O escalonador emprega uma abordagem estocástica com aprendizado por reforço, na qual são construídas correlações estatísticas entre o usuário, o comportamento das suas aplicações e o ambiente de execução.

Instanciação otimizada das aplicações: o escalonador carrega nos *EXEHDAnodos* (móveis ou fixos) um conjunto mínimo de componentes de software que garantam a execução da aplicação (valendo-se do perfil do usuário), caracterizando uma estratégia *push* de operação. Os outros componentes, se necessários, serão solicitados sob demanda, caracterizando uma estratégia *pull* de operação.

Instanciação antecipada das aplicações: o processo de instanciação começa no momento em que o usuário efetiva sua autenticação na *EXEHDABase*, antes de solicitar a execução de aplicações. Neste caso, adota-se uma estratégia *push* de disseminação de componentes de software e informação. Esta instanciação também pode ocorrer com uma antecipação ainda maior, tendo por referência uma expectativa de roteiro de mobilidade do usuário já consolidada. Antecipar o tráfego na parte estruturada da rede (com conexão física) é uma opção da arquitetura proposta para aumentar o desempenho global da aplicação móvel, e conseqüentemente reduzir o tempo de espera/conexão do usuário do segmento de rede com suporte à mobilidade (conexão sem fio).

4 Aplicando a Adaptação Multinível Colaborativa ISAM

Esta seção tem por objetivo instanciar o emprego da estratégia de adaptação do ISAM. A avaliação da Adaptação Multinível Colaborativa será feita usando aplicações distribuídas voltadas para o aumento de desempenho.

A resolução de problemas numericamente intensivos é, em geral, computacionalmente dispendiosa, exigindo um elevado tempo de processamento. Muitos destes problemas apresentam características que facultam dividir o que deve ser processado em *Unidades de Trabalho* (UTs), as quais podem ser calculadas de forma independente.

Objetivando reduzir o tempo para solução desse tipo de problema, foi implementada uma ICP (*Internet Computing Platform*) sob a arquitetura ISAM, denominada UniCluster. A sua codificação em ISAMadapt, empregando o Holoparadigma foi mapeada para Java [26].

4.1 UniCluster

O UniCluster é composto por *servidores* distribuídos, alocando UTs, associados a um conjunto de *clientes* encarregados de realizar o processamento sobre as mesmas. A topologia de interconexão dos servidores é determinada pelo administrador do sistema. O principal critério de visibilidade entre os servidores é a velocidade da interconexão de rede existente. Os principais componentes da arquitetura do UniCluster são:

- servidores - compostos de dois fluxos de execução: (*i*) um dedicado a atender os pedidos de informações sobre o estado atual do sistema, e outro (*ii*) tanto para atender às solicitações de tarefas para processamento, como para recebimento de resultados parciais;
- clientes - a proposta UniCluster contempla a existência de três tipos de programas clientes:
    - *processor clients*: clientes processadores que executam os algoritmos de interesse do usuário e trabalham sobre as UTs, retornando resultados ao final do processamento de cada uma;
    - *loader clients*: clientes que submetem aplicações e seus parâmetros para execução;
    - *status clients*: clientes que buscam informações sobre o estado do sistema (processamentos em andamento, o que já foi realizado, etc.), bem como resultados finais de problemas já resolvidos.

Para permitir que o usuário possa iniciar e acompanhar suas execuções de qualquer lugar, sem precisar instalar software específico para tal, os *loader clients* e os *status clients* foram dotados de interface WEB. Isto faculta que, de qualquer navegador HTML, o usuário possa acompanhar os processamentos de seus problemas. Eventos significativos também podem ser notificados por e-mail. O estudo de caso deste artigo é baseado em um dos problemas resolvidos no UniCluster, denominado VIGAPI.

4.2 VIGAPI: Um Problema de Engenharia Civil

A análise de vigas e pilares de concreto normalmente é feita utilizando-se valores médios para as propriedades dos materiais e valores nominais para as dimensões da estrutura. Contudo, esses dados sempre apresentam desvios em relação aos valores reais. Para avaliar os efeitos destas incertezas sobre a resposta estática da estrutura, é empregado o método de Monte Carlo. Os dados de entrada de cada análise são gerados aleatoriamente, de acordo com suas distribuições de probabilidade. As análises são repetidas inúmeras vezes, obtendo-se diversas respostas, que são armazenadas. Um processamento estatístico sobre os dados permite determinar a média, o desvio padrão e estimar a distribuição de probabilidade de deslocamentos, deformações e tensões. Esta estratégia foi utilizada na concepção da aplicação VIGAPI [21] .

Particularmente no caso da aplicação VIGAPI, o ganho a ser obtido com o uso da estratégia multinível colaborativa em redes de nodos heterogêneos, tem origem nas barreiras de sincronização que ocorrem durante as execuções paralelas. No momento de uma sincronização, uma tarefa que esteja em execução em um nodo lento, pode postergar a continuidade do processamento distribuído.

Por sua vez, aplicações baseadas em métodos estatísticos, semelhantes à VIGAPI, facultam serem decompostas em Unidades de Trabalho (UTs) com custo computacional diferenciado. A estratégia adaptativa ficou com a seguinte organização:

- no nível de aplicação: foram especificados três níveis de custo computacional para as UTs. Para expressar essas informações foram empregadas as primitivas de adaptação do ISAMadapt;
- no nível de arquitetura do sistema: o escalonador do gerente da execução, (*Servidor*) utilizando as especificações do usuário, entrega para os processadores (*processor clients*) as UTs compatíveis com o poder computacional do nodo que o aloja (os nodos processadores foram agrupados em três conjuntos - veja coluna "Grupos" na tabela 1).

4.3 Avaliando os Resultados

Para avaliar o comportamento da arquitetura adaptativa do ISAM, processou-se diversas vezes o problema de Engenharia Civil, empregando o UniCluster no ambiente apresentado na tabela 1. Este ambiente se caracteriza por uma elevada heterogeneidade do poder computacional dos equipamentos envolvidos (coluna "Poder Relativo" na tabela 1), condição comum em uma ICP. Os equipamentos estão interligados por uma rede Ethernet de 10 Mbps. As medidas foram feitas com os equipamentos e a infra-estrutura de rede, dedicadas ao experimento. A configuração contemplou um servidor e cinco clientes. O baixo custo computacional do servidor permitiu que o mesmo fosse executado juntamente com um dos clientes de processamento.

Tabela 1. Ambiente de Execução Utilizado

Nodo	Nome do Host	Plataforma	Sist. Op.	Poder Relativo[1]	Grupos 3 níveis	Grupos 2 níveis
1	guaiaca	Sun Ultra 10 – 128 Mbytes RAM	SunOS 5.8	6,80	1	1
2	spica	Sun Ultra 5 – 192 Mbytes RAM	SunOS 5.8	5,44	2	1
4	poncho	Sun SPARCstation 20 – 128 Mbytes RAM	SunOS 5.8	3,40	2	2
3	jacui	Sparc Enterprise 150 – 128 Mbytes RAM	SunOS 5.5.1	2,42	3	2
5	adhara	Sparc Statiton 4 – 128 Mbytes RAM	SunOS 5.8	1,00	3	2

A figura 6 compara os tempos de execução para a aplicação VIGAPI utilizando ou não a estratégia adaptativa. Os processamentos foram realizados para diferentes combinações possíveis dos dados de entrada: número de simulações, número de UTs, dimensões da peça de concreto, margens de segurança, etc. A linha diagonal representa tempos de execução iguais, e os valores registrados (em segundos) são decorrentes da média de oito execuções. O aumento de desempenho da execução distribuída da aplicação VIGAPI com o uso da Adaptação Multinível Colaborativa variou de 10% a 50%, atingindo um valor médio de 30%.

Em função dos custos de comunicação inerentes a toda proposta de ICP, o UniCluster consegue atender aplicações distribuídas de granulosidade[2] média e elevada. O UniCluster, como foi concebido, é bastante apropriado para problemas do tipo mestre/escravo (um servidor atendendo múltiplos agentes de cálculo), como a aplicação VIGAPI. Porém, este também pode ser aplicado a problemas com um nível moderado de interdependência, situação em que cresce o número de barreiras de sincronização necessárias. O crescimento no número de barreiras de sincronização potencializa o ganho com o uso da estratégia de adaptação multinível colaborativa.

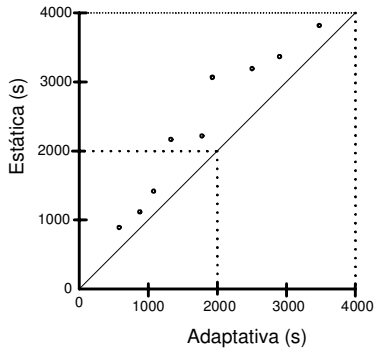


Figura 6. Tempos de Execução das Estratégias Estática e Adaptativa

A tabela 2 e a figura 7 registram o comportamento do desempenho na execução de uma instância do problema VIGAPI sob o ambiente da tabela 1, à medida que variam os níveis de adaptação. Entende-se por nível de adaptação, o número de classes de UTs (custo computacional) e de nodos processadores (em termos de poder computacional – veja tabela 2). No nível 1 todos os processadores são considerados iguais, e conseqüentemente as UTs também o são.

[1] Poder computacional relativo dos hosts, medido utilizando *benchmark* com comportamento semelhante à aplicação VIGAPI (manipulação de funções matemáticas e uso intensivo de processador).

[2] Granulosidade: razão entre o tempo de processamento da função a ser paralelizada e o tempo de comunicação necessário para o envio e o recebimento dos dados.

Tabela 2. Desempenho & Níveis de Adaptação

	5 Uts		10 Uts	
Níveis de UTs	Média (s)	Desvio Padrão (s)	Média (s)	Desvio Padrão (s)
1	3200	5,3	1970	2,7
2	2930	3,5	1890	2,3
3	2120	3,0	1510	2,1

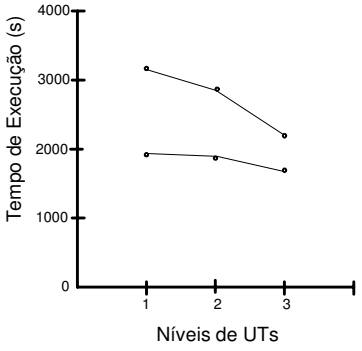


Figura 7. Desempenho & Níveis de Adaptação

O problema foi resolvido para dois níveis de granulosidade (a granulosidade com 5 UTs é o dobro da atingida com 10 UTs). O uso da estratégia adaptativa melhorou o desempenho nos dois níveis.

Pode-se observar que a redução da granulosidade melhorou o desempenho, reduzindo as perdas com a barreira de sincronização. Porém, é importante registrar que a redução da granulosidade é limitada tanto pelos custos de comunicação e de gerência da distribuição, como pela natureza do problema a ser resolvido. A figura 7 mostra que o uso da estratégia adaptativa aproximou os resultados atingidos com as duas granulosidades. Melhorar a possibilidade de trabalhar com granulosidades mais elevadas é particularmente conveniente para propostas que contemplem redes *wide-area*, nas quais os custos de comunicação sempre devem ser considerados.

Avaliando os testes, duas constatações se destacam:

comportamento adaptativo do paralelismo**:** o UniCluster permite a alteração do número de processadores alocados para uma aplicação, durante sua execução. Decorrem desta característica dois aspectos: (*i*) uma melhor possibilidade de aproveitamento da arquitetura, pois os nodos não ficam presos à determinada aplicação à medida que oscila a sua exigência de processadores, (*ii*) a possibilidade de não precisar aguardar que a aplicação atual encerre completamente, para direcionar os processadores em uso para novas aplicações. Isto diferencia o UniCluster de propostas bastante disseminadas como o PVM e o MPI [5]. Esta característica é uma decorrência da estratégia anônima e assíncrona de entrega das UTs (uso de *blackboard*: principal estratégia da arquitetura ISAM para comunicação/sincronização entre processos);

aumento do paralelismo médio: registrou-se que o emprego da adaptação multinível colaborativa para gerenciar a entrega seletiva de carga, considerando o poder computacional dos processadores envolvidos, conseguiu reduzir os custos com equipamentos parados nas barreiras de sincronização. Deste aumento é que resultaram as melhorias de desempenho observadas nas figuras 6 e 7.

5 Trabalhos Relacionados

Existem diversos trabalhos relacionados com a premissa de empregar a Internet como infra-estrutura para aplicações distribuídas. Sistemas como o Condor [13] são voltados para aplicações de alto desempenho em *clusters* de estações de trabalho. Diferentemente do ISAM, utilizam um mecanismo central para disparar processos. O projeto Globus [11] disponibiliza uma "grade de recursos computacionais" [12] integrando equipamentos heterogêneos em um único sistema. De forma similar à proposta ISAM, ele contempla uma estrutura escalável e distribuída para o gerenciamento de recursos. Apesar de conter um módulo específico para o controle de aplicações (GEM – *Globus Executable Management Service*), a atual versão trata as aplicações como um único módulo executável, ao invés de uma coleção de componentes que podem ser parcial e dinamicamente instanciados, como na arquitetura ISAM.

Por sua vez, sistemas como Globe [23], Legion [2] e WebOs [28], apesar de suportarem diferentes níveis de configuração, não consideram a adaptabilidade e a configuração automática do ambiente de execução como uma questão central. Por outro lado, as pesquisas em aplicações móveis adaptativas focalizam várias facetas da mobilidade e podem ser agrupadas em três categorias: (*i*) monitoramento de recursos, em especial recursos da rede [29][7][1]; (*ii*) projeto de aplicações móveis específicas [14][4][25]; (*iii*) *toolkits* para o desenvolvimento de aplicações [1][27][20]. Esses sistemas tratam de aspectos específicos do ambiente móvel. Diferente destes sistemas, a arquitetura ISAM é mais abrangente, propondo a integração do tratamento da mobilidade de hardware e de software. ISAM oferece o paradigma e a linguagem para o desenvolvimento das aplicações, e também o sistema de execução que monitora o contexto e fornece mecanismos de adaptação às alterações contextuais.

6 Considerações Finais

A arquitetura ISAM trata o problema da adaptação no cenário das aplicações distribuídas em escala global (*Pervasive Computing, MetaComputing*) através de um modelo multinível colaborativo. A aplicação estabelece políticas de adaptação, e tem um comportamento reativo à variação nas condições do seu contexto. Porém, diferentemente da maioria das outras propostas, esta reação é negociada com o ambiente de execução, que gerencia simultaneamente todas as aplicações em execução. Neste artigo foi resumida a proposta ISAM, e caracterizada a sua estratégia de Adaptação Multinível Colaborativa. Esta estratégia foi utilizada no desenvolvimento de um ambiente de ICP (*Internet Computing Plataform*). O UniCluster executa aplicações distribuídas numericamente intensivas e proporciona melhoria de desempenho.

O desenvolvimento de aplicações em ambientes distribuídos *wide-area*, nos quais os recursos podem estar em estados não previsíveis, é uma tarefa complexa. A adição da mobilidade de hardware neste contexto traz um conjunto de novos requisitos e desafios para a produção de software, e cria demanda por novos tipos de aplicações. Esta perspectiva exige uma capacidade de adaptação maior que a apresentada pelos Sistemas de Gerenciamento de Recursos atuais. Exatamente este é o escopo de pesquisa explorado pelo projeto ISAM.

Referências Bibliográficas

[1] A. Dey; G. Abowd. *Towards a Better Understanding of Context and Context-Awareness*. Proceedings of Conference on Human Factors in Computing Systems (HCI2000), 2000.

[2] A. Grimshaw et al. The *Legion Vision of a World-Wide Virtual Computer*. Communications of the ACM. New York, v.40, n.1, 1997.

[3] A. Yamin, I. Augustin, J.V. Barbosa, C.F.R. Geyer. *Exploring the Scheduling in Mobile Distributed Applications Performance*. Proc. Workshop in High Performance Computational Systems, Goias, Brazil, 2001.

[4] B. Noble. *System Support for Mobile, Adaptive Applications*. IEEE Personal Computing Systems, 7(1), Feb. 2000.

[5] D. Culler; J.P Singh. *Parallel Computer Architecture: a hardware and software approach***.** Morgan Kaufmann. 1999. 479p.

[6] E. Araujo; I. Augustin; A. Yamin; L. Silva; C.F.R. Geyer. *Uma Proposta de Monitoração para Visualização de Aplicações Distribuídas Java*. Jornadas Chilenas de Computación 2001. V Workshop en Sistemas Distribuídos y Paralelismo. Chile. 5-9 Nov. 2001.

[7] G. Welling;B.R. Badrinath. *An Architecture for Exporting Environment Awareness to Mobile Computing Applications*. IEEE Transactions on Software Engineering. v. 24, n.5. 1998.

[8] G.G.H. Cavalheiro; Y. Denneulin; J.L. Roch. *A General Modular Specification for Distributed Schedulers*. Proceedings of Europar'98. Southampton, Springer Verlag, LNCS 980. 1998.

[9] I. Augustin, A . Yamin, C. Geyer. *Requisitos para o Projeto de Aplicações Móveis Distribuídas*. VIII CACIC Congreso Argentino de Ciencias de la Computación, Argentina, Oct, 2001.

[10] I. Augustin, A. Yamin, J. Barbosa, C. Geyer. *Towards a Taxonomy for Mobile Applications with Adaptive Behavior*. International Symposium on Parallel and Distributed Computing and Networking (PDCN02), Innsbruck, Austria, Feb., 2002. Accepted for publication.

[11] I. Foster; C. Kesselman. *The Globus Project: A Status Report*. Proceedings of the IPPS/SPDP - Heterogeneous Computing Workshop. 1988.

[12] I. Foster; C. Kesselman: Editors. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers. San Francisco. 1999.

[13] J. Basney and M. Livny. *Managing Network Resources in Condor*. Proceedings of the Ninth IEEE Symposium on High Performance Distributed Computing (HPDC9), Pittsburgh, Pennsylvania, August 2000, pp 298-299.

[14] J. Jing; K. Huff. *Adaptation for Mobile Workflow Applications*. Proceedings of Workshop on Modeling and Simulation in Wireless Systems. Montreal, Canada. Jul. 1998.

[15] J.L.V. Barbosa, A.C. Yamin, P.K. Vargas, Augustin I., C.F.R. Geyer. *Holoparadigm: a Multiparadigm Model Oriented to Development of Distributed Systems*. International Conference on Parallel and Distributed Systems (ICPADS), New York, IEEE Press, 2002.

[16] J.L.V. Barbosa, C.F.R. Geyer. *A Multiparadigm Language Oriented to Distributed Software Development*. V Brazilian Symposium of Programming Languages. (SBLP), 2001.

[17] L. Silva; A. Yamin; I. Augustin; J.L.V. Barbosa; C.F.R. Geyer. *Mecanismos de Suporte ao Escalonamento em Sistemas com Objetos Distribuídos Java*. VIII CACIC Congreso Argentino de Ciencias de la Computación. Santa Cruz, Argentina. Oct, 2001.

[18] M. Baker, R. Buyya, D. Laforenza. *Grids and Grid Technologies for Wide-Area Distributed Computing*. Technical Report: 2001/92, Monash University, may 2001.

[19] M. Maheswaran. *Quality of Service Driven Resource Management Algorithms for Network Computing*. The International Conference on Parallel and Distributed Processing Techniques and Applications. PDPTA, June 1999.

[20] M. Ranganathan; A. Acharya; J. Saltz. *Sumatra: a Language for Resource-aware Mobile Programs*. Mobile Objects Systems: Towards the Programable Internet: Springer-Verlag Publisher, Serie Lecture Notes on Computer Science. v. 1222. Apr. 1997.

[21] M. Real; A. Campos. *Aplicação do método de Monte Carlo no Estudo de Vigas e Pilares de Concreto Armado*. Revista Teoria e Prática na Engenharia Civil, Rio Grande, n.2, p.35-44, Maio 2001.

[22] M. Satyanarayanan. *Pervasive Computing: Vision and Challenges*. IEEE Personal Communications, 2001.

[23] M. Steen et al. *Globe: A Wide-Area Distributed System*. IEEE Concurrency. New York, v.7, n.1. 1999.

[24] N. Davies, A. Friday, S. Wade, G. Blair. *Limbo: a Tuple Space Based Platform for Adaptive Mobile Applications*. Proceedings of the International Conference on Open Distributed Processing/Distributed Platforms (ICODP/ICDP´97), Toronto, Canada, 27-30, May 1997.

[25] N. Davies, K. Cheverst, K. Mitchell. *A Friday, Caches in the air: Disseminating Tourist Information in the GUIDE System*. Proceedings of 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleand, USA, 1999.

[26] R. Real; A. Yamin; M. Real; N. Duarte; O. Salvador. *UNICLUSTER: uma proposta de ICP voltada para PAD*. ERAD 2002. SBC. São Leopoldo, 2002.

[27] T. Kunz; J.P. Black. *An Architecture for Adaptive Mobile Applications*. Proceedings 11th International Conference on Wireless Communications. Alberta, Canada. Jul. 1999.

[28] T. Vahdat et al. *WebOS: Operating System Services for Wide Area Applications*. Proceedings of the Seventh Symposium on High Performance Distributed Computing. 1998.

[29] T. DeWitt et al. *ReMoS: A Resource Monitoring System for Network-aware Applications*. Technical Report. CMU-CS-97-194. Carnegie Mellon University. Dec. 1997.

[30] V. Bharchavan et al. The *Timely Adaptive Resource Management Architecture*. IEEE Personal Communications Magazine. v.5, n. 4, 1998.

Construindo uma Fábrica de Software: da Concepção às Lições Aprendidas

Vivianne da Nóbrega Medeiros
Carlos Andreazza Rego Andrade
Eduardo Santana de Almeida
Jones Albuquerque
Silvio Meira
Universidade Federal de Pernambuco, Centro de Informática,
Recife, Brasil, 50732-970
{vnm, cara, esa2, joa**,** slrm}@cin.ufpe.br

e

Eduardo Santana de Almeida
Silvio Meira
C.E.S.A.R. - Centro de Estudos e Sistemas Avançados do Recife
Recife, Brasil, 50030-390
silvio@cesar.org.br

Abstract

In recent years, there have been a number of experiences with software factories reported in the literature which, although dense in a number of ways, have been lacking in what regards the discussion of the stages of definition and setting up of factories themselves. This paper discusses a number of issues related to the conception, implementation and improvement of a software factory and, as a result of a real life experiment, also points to a number of lessons learned, which can very likely be replicated within similar contexts.

Keywords: Software Engineering, Software Factory

Resumo

Pesquisas e esforços envolvendo fábricas de software têm sido apresentados ao longo dos anos na literatura. Entretanto, ainda existe uma carência de relatos de experiências envolvendo as etapas necessárias para sua definição e construção. Assim, este artigo apresenta experiências com uma fábrica de software descrevendo desde a fase de definição, até as lições aprendidas durante todo o processo, oferecendo uma contribuição significativa neste seguimento.

Palavras-chave: Engenharia de Software, Fábrica de Software

1. Introdução

O desempenho cada vez mais otimizado das fábricas industriais clássicas, a consolidação das técnicas de engenharia de software, juntamente com o refinamento dos ambientes de desenvolvimento e o surgimento de novos ambientes de projeto e suporte integrados têm feito com que, cada vez mais, esforços sejam despendidos no sentido de realizar o conceito de Fábrica de Software [7]. Esse conceito simboliza uma desejada mudança de paradigma da produção de software focada no trabalho intensivo, para um estilo mais focado no capital, onde investimentos substanciais podem ser feitos sob um nível de risco aceitável.

As fábricas clássicas, onde as pessoas atuam como máquinas na realização de tarefas pré-determinadas, não é um modelo nem desejável, nem correto para fábrica de software. No contexto de software, a analogia com a fábrica pode ser aplicada apenas aos objetivos da produção baseada no estilo industrial, e não na sua implementação. A manufatura de software envolve pouca ou nenhuma produção tradicional: "todo sistema é único; apenas partes individuais podem aparecer repetidamente em mais de um sistema" [7].

No tocante ao conhecimento disseminado sobre fábricas de software, diversos autores [5, 6, 8, 13, 15] consideram que, muito se tem discutido sobre aspectos tecnológicos envolvendo o desenvolvimento dos produtos, enquanto que a ênfase dada sobre os processos envolvidos numa fábrica de software, principalmente, processos de definição e planejamento são negligenciados.

Neste trabalho, apresenta-se uma análise sobre as questões que devem ser consideradas durante o processo de elaboração, gestão e implantação de uma fábrica de software. Adota-se uma forma de organização para a fábrica em questão e aplica-se a mesma em duas aplicações, desenvolvidas em ambiente acadêmico, para validar a estrutura proposta, resultando no detalhamento das lições aprendidas com essas experiências, que podem ser replicadas dentro de contextos similares.

O artigo está organizado em quatro seções. A primeira seção contém esta introdução. A Seção 2 descreve o processo de criação da fábrica propriamente dito e sua implantação, sendo dividida em três subseções que detalham as lições extraídas, respectivamente, da fase de concepção, do projeto piloto e do projeto real utilizado; a Seção 3 exibe os trabalhos relacionados e, finalmente, a Seção 4 apresenta as conclusões e perspectivas de trabalhos futuros.

2. Construção de uma Fábrica de Software

Nesta seção, detalham-se as etapas percorridas durante a elaboração de uma fábrica de software e sua conseqüente implantação, exibindo assim, resultados coletados com as experiências práticas vivenciadas. O modelo de fábrica proposto foi aplicado a um projeto piloto com a finalidade de gerar refinamentos e melhorias no processo adotado; em seguida, um projeto mais amplo foi desenvolvido para validar e verificar a viabilidade da estrutura definida.

2.1 Concepção

A fábrica foi definida em função do contexto no qual estava inserida. A idéia consistia em adotar um processo de desenvolvimento de software baseado em componentes com métodos para avaliar o progresso, técnicas para acompanhar andamento das atividades e ferramentas para otimizar a construção dos artefatos, tendo produtos desenvolvidos em um ambiente acadêmico. Devido à necessidade de incrementar a produtividade e a qualidade dos seus processos e produtos, assim como reduzir os custos e o tempo de desenvolvimento, a proposta de solução considerava dois importantes pontos: melhorar a eficiência dos processos e reutilizar os artefatos previamente construídos.

Dessa forma, durante esta etapa de concepção, que se estendeu por quatro (4) semanas, atividades relacionadas à forma de organizar a estrutura geral de uma fábrica de software foram consideradas, tendo em vista os objetivos já mencionados. As principais decisões tomadas para possibilitar a construção desse ambiente foram:

- **Perfis funcionais** e as respectivas **atividades** a serem desempenhadas;
- **Metodologia de desenvolvimento de software** a ser utilizada incluindo artefatos e **métricas**;
- Definir um **plano de processos** descrevendo as atividades e relacionando-as com os artefatos e perfis funcionais responsáveis pela execução das mesmas;
- **Material de instrumentação** necessário.

A seguir, as soluções adotadas pela fábrica proposta para contemplar os aspectos acima citados são apresentadas.

2.1.1 Perfis Funcionais

A classificação estabelecida para os perfis funcionais, juntamente com as responsabilidades de cada cargo:

- **Gerente de Negócios:** prospecção do mercado e venda dos serviços.

- **Gerente de Projeto***:* gerenciamento dos riscos e das atividades em desenvolvimento devendo dimensionar e alocar os recursos necessários para realização das tarefas de forma satisfatória, além de interagir com o cliente e o gerente de negócios.
- **Analista de Sistemas***:* levantamento de requisitos, análise, definição da arquitetura e documentação do sistema a ser desenvolvido.
- **Analista de Qualidade***:* revisão dos artefatos gerados, controle de mudanças, bem como a definição e validação da qualidade e acurácia do processo utilizado pela fábrica.
- **Engenheiro de Software**: implementação do sistema conforme as especificações e documentação, seguindo o processo de desenvolvimento definido.
- **Engenheiro de Testes***:* desenvolvimento, validação e execução de testes de software com o intuito de assegurar a qualidade e acurácia do software produzido.
- **Líder de Equipe**: coordenação e atribuição de tarefas dentro de um grupo específico, relatando periodicamente ao gerente de projetos o andamento das atividades.

2.1.2 Metodologia de Desenvolvimento

A metodologia de desenvolvimento baseou-se no Rational Unified Process (RUP) [9] e no Project Management Body of Knowledge (PMBOK) [4]. A essência do RUP consiste em um ciclo de desenvolvimento iterativo e incremental, assim, ao longo do tempo são executadas várias iterações e cada uma delas gera versões contemplando um número maior de funcionalidades. Por sua vez, o PMBOK é aceito como um padrão para gerenciamento de projetos e descreve práticas tradicionais e largamente aplicadas para entendimento e execução das atividades relativas a gestão do projeto, incluindo objetivos, responsáveis e participantes de cada processo.

A metodologia de desenvolvimento adotada na fábrica está dividida nas seguintes fases:

- **Comercial**: definição do projeto, de acordo com as necessidades levantadas junto ao cliente, além do levantamento de estimativas de custo e esforço baseando-se na técnica de Pontos de Caso de Uso Ajustados [10]. A intenção é unificar o entendimento do produto a ser desenvolvido e fornecer uma estrutura com previsões razoáveis de recursos, custos e prazos; dados históricos devem ser coletados para utilização em estimativas de projetos futuros, aumentando assim as chances de sucesso.
- **Planejamento e Gerenciamento***:* elaboração do plano do projeto (plano de trabalho, riscos, acompanhamento e controle), estimação de prazos e execução das atividades recorrentes para avaliar o progresso, coletando as métricas de desempenho ($\mu_{\text{concluído}}$, $\mu_{\text{atraso médio}}$ e $\mu_{\text{novas atividades}}$) definidas em [12]. O gerente de projetos terá atribuições e responsabilidades descritas no PMBOK. A meta é fornecer uma linha básica de estruturação e delimitação do trabalho, devendo produzir ações como: comunicar o escopo e os recursos a todos os envolvidos; definir riscos e sugerir técnicas para evitá-los, ou minimizá-los; elaborar cronograma com divisão de trabalho e dependência entre as atividades; gerenciar mudanças; planejar o acompanhamento das tarefas existentes e registrar o monitoramento através de relatórios, a fim de checar o planejado com o realizado efetivamente. Enfim, desenvolve-se mecanismos para avaliar o progresso, organizar o pessoal que desenvolverá o produto, além de rastrear e controlar o projeto e mudanças que por ventura apareçam.
- **Desenvolvimento de Componentes***:* definição do problema, especificação, projeto e implementação dos componentes. Esta fase segue uma abordagem definida em [1], considerando a reutilização como fator primordial e, assim, objetivando mais qualidade e redução nos custos de desenvolvimento, testes, documentação e manutenção. Inicialmente, a abordagem parte dos requisitos do domínio do problema e produz os componentes implementados numa linguagem orientada a objetos. Uma vez implementados, o engenheiro de software desenvolve as aplicações reutilizando os componentes previamente construídos.
- **Testes e Validação***:* elaboração de testes para validar os artefatos previamente construídos. Representa a última oportunidade de detectar erros antes do software ser distribuído aos usuários e tem como principais objetivos: planejar os testes que devem ser executados em cada iteração; verificar a correta integração entre todos os componentes do software; averiguar se todos os requisitos do sistema foram corretamente implementados; executar vários testes para comparar o resultado dos mesmos com os parâmetros definidos como esperados, a fim de produzir uma indicação da qualidade e da confiabilidade do software; além de realizar os testes de aceitação junto ao cliente.

2.1.3 Plano de Processos

O plano de processos detalha cada fase da metodologia, mostrando o fluxo de atividades a serem executadas, os artefatos que devem ser produzidos e os perfis funcionais encarregados da geração de tal material. A Tabela 1 mostra as atividades descritas no plano de processos da fábrica.

Tabela 1. Fluxo de atividades definidas no plano de processos

	Atividades	Responsável	Artefatos
Comercial	Levantar Necessidades do Cliente	Gerente de Negócios	- Ata de Reunião
	Elaborar Proposta Técnica	Líder de Equipe	- Proposta Técnica
	Estimar Esforço do Projeto	Gerente de Negócios	- Planilha de Estimativa de Esforço
	Elaborar Proposta Comercial	Gerente de Negócios	- Proposta Comercial
Planejamento e Gerenciamento	Elaborar Plano de Projetos Preliminar	Gerente de Projeto	- Plano de Projeto Preliminar
	Definir o Controle do Projeto	Gerente de Projeto Equipe de trabalho	- Work Breakdown Structure (WBS) Cronológico - Definição do Plano de Acompanhamento e Controle - Plano de Gerenciamento de Impactos - Plano de Gerenciamento de Configuração - Plano de Comunicação
	Acompanhar e Gerenciar o projeto	Gerente de Projeto Analista de qualidade	- Work Breakdown Structure (WBS) Cronológico atualizado - Plano de Acompanhamento e Controle atualizado - Plano de Gerenciamento de Impactos atualizado - Plano de Gerenciamento de Configuração atualizado - Formulário de Controle de Mudanças
	Comunicar através de Reuniões Periódicas	Gerente de Projeto Líderes de equipes	- Ata de Reuniões
	Validar o Projeto	Analista de Qualidade Gerente de Negócios Gerente de Projeto Líderes de equipes	- Formulário de Validação do Cliente - Documento Avaliando o Processo Adotado
Desenvolvimento de componentes	Definir Problema	Analista de sistema	- Mind-maps - Modelo de colaboração - Modelo de casos de uso - Documento de Requisitos
	Especificar Componentes	Analista de sistema	- Modelo de tipos - Framework de modelos - Aplicação do framework - Refinamento dos diagramas de iteração
	Projetar Componentes	Analista de sistema	- Modelo de classes - Refinamento dos diagramas de iteração
	Implementar Componentes	Engenheiro de software	- Código gerado
Testes e Validação	Elaborar Plano de Testes	Engenheiro de testes	- Plano de Testes
	Implementar Testes	Engenheiro de software	- Componentes de Teste
	Executar Testes	Engenheiro de testes	- Registro dos resultados
	Avaliar Testes	Engenheiro de testes	- Relatório de Avaliação de Testes
	Executar Testes de Aceitação	Usuário validador	- Observações do validador

Cada atividade descrita no plano de processos expõe as informações de quais são os seus objetivos, insumos necessários à sua realização (entradas), seqüência de passos a serem executados, resultados gerados (saídas) e perfis responsáveis pela condução da mesma. Modelos foram elaborados para cada documento definido, com o intuito de servir como roteiro durante a construção do artefato e, assim, diminuir o esforço de criação dos mesmos, garantir a presença de informações relevantes, além de facilitar a verificação de aderência à metodologia.

2.1.4 Material de Instrumentação

As tecnologias empregadas pela fábrica foram definidas de acordo com as categorias:

- *Ferramentas de desenvolvimento e modelagem;*
- *Ferramentas para relatar bugs durante o desenvolvimento;*
- *Ferramentas de gerenciamento de projetos;*
- *Ferramentas para comunicação entre os participantes e disseminação do conhecimento; e*
- *Sistema gerenciador de banco de dados.*

2.2 Calibração de Fábrica: Projeto Piloto

2.2.1 Contexto

Com o objetivo de validar e ajustar a fábrica definida, um projeto piloto foi realizado em cinco (5) semanas. Este projeto não foi complexo, já que o mais importante seria o "aculturamento" dos participantes nos processos, atividades, artefatos e tecnologias envolvidas, como também possibilitar a identificação de possíveis inconsistências durante o processo e evitar o seu posterior impacto na fase de produção, ou seja, a finalidade principal desta etapa seria ambientar os envolvidos com a estrutura utilizada, resultando em sugestões de mudanças quando necessário.

Os participantes foram alocados nos perfis apresentados na Seção 2.1.1, considerando a conveniência e o conhecimento de cada indivíduo. Entretanto, todos desempenharam mais de um perfil, a fim de evitar a ociosidade e melhorar o desempenho geral da fábrica. Os envolvidos formavam um total de treze pessoas e constituía-se de alunos dos cursos de Mestrado e Doutorado do Programa de Pós-Graduação em Ciência da Computação, da Universidade Federal de Pernambuco, que cursavam uma disciplina de Engenharia de Software, além dos dois instrutores da disciplina que desempenhavam o papel de clientes.

2.2.2 Execução

O projeto constituiu-se no desenvolvimento de um sistema de alocação de disciplinas, professores, salas e horários, para auxiliar os funcionários da Secretaria de Pós-Graduação em Ciência da Computação. As funcionalidades contempladas estão ilustradas num diagrama de casos de uso na Figura 1 e consistiam basicamente da manutenção (cadastro, atualização e remoção) de salas, disciplinas, horários, professores e alocações.

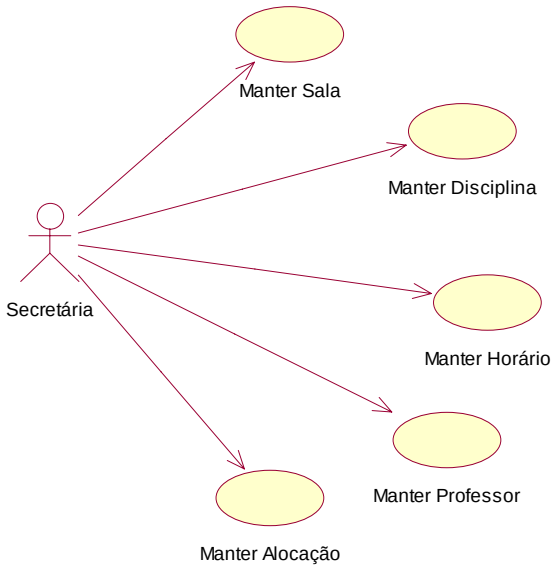


Figura 1: Diagrama de casos de uso desenvolvidos no Projeto Piloto

As etapas estabelecidas no plano de processo foram realizadas e os relatórios gerados semanalmente mostrou-se um fator essencial para o bom acompanhamento do progresso das atividades. As métricas de desempenho coletadas foram às definidas em [12] e contemplam o escopo de tempo definido para a avaliação, apresentando-se sob as seguintes variáveis:

- $\mu_{\text{concluído}}$:o quanto uma determinada equipe concluiu das atividades que estavam inicialmente planejadas;
- $\mu_{\text{atraso médio}}$:o percentual de atraso médio por atividade; e
- $\mu_{\text{novas atividades}}$:o percentual médio de tempo gasto com atividades não planejadas que alocam o pessoal da equipe, prejudicando o planejamento inicial e o desempenho. A partir desses dados, o cliente mantinha-se informado a respeito do trabalho desenvolvido e dispunha de meios práticos para avaliar a evolução do projeto.

De modo a garantir a qualidade do software piloto desenvolvido e os futuros sistemas produzidos pela fábrica, elaborou-se uma auditoria interna com o intuito de catalogar as boas práticas, desvios no processo, oportunidades de aperfeiçoamento, entre outros. Assim, foi verificado se o projeto estava seguindo a política de desenvolvimento estabelecida, a capacidade e necessidade de treinamento dos funcionários, atividades de gerência de requisitos, além de planejamento e acompanhamento de projetos e gerência de configuração. Ações de melhoria contínua, baseadas nos resultados analisados, puderam ser endereçadas.

A quantidade de horas utilizadas por fase durante a execução do piloto foram registradas e são apresentados na Figura 2.

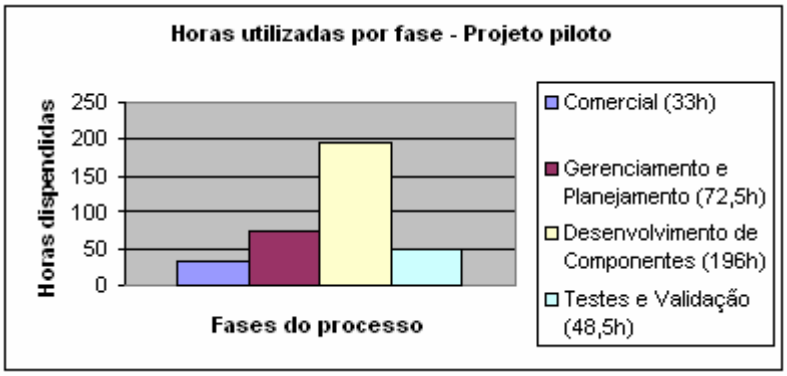


Figura 2: Horas utilizadas por fase no piloto

2.2.3 Análise

O projeto piloto foi realizado sem maiores problemas e o sistema foi implementado com o escopo requisitado e o prazo previsto. Esse projeto foi de extrema importância, pois permitiu aos participantes adquirir experiência na metodologia definida, verificar a adequação dos participantes em relação aos perfis funcionais e experimentar todo o processo operacional de uma fábrica de software. Entretanto, alguns pontos apresentados durante a realização do projeto piloto merecem ser destacados, a saber:

i. Sobrecarga de Trabalho*:* durante a execução do projeto, houve sobrecarga de trabalho relacionado à equipe de desenvolvimento. Este fato ocorreu, principalmente, pela baixa interseção dos horários de disponibilidade entre os membros da fábrica por se tratarem de alunos de Pós-Graduação com outras responsabilidades além da disciplina de Engenharia de Software, levando, muitas vezes, a falhas de comunicação;

ii. Alocação de Perfis*:* alguns participantes não estavam bem alocados em relação ao perfil funcional que se encontravam causando, deste modo, alguns atrasos de cronograma;

iii. Instrumentação*:* a ferramenta CASE adotada inicialmente para modelagem do sistema [2] não estava atendendo da forma esperada, uma vez que não permitia o trabalho colaborativo entre os participantes, atrasando, de certa forma, as tarefas dos membros da equipe de Analistas de Sistemas;

iv. Avaliação do processo*:* ao fim do piloto, o processo como um todo foi avaliado por todos os componentes da fábrica, através de questionários onde a eficácia do mesmo foi analisada. Durante uma reunião, após o término do piloto, foi feita uma avaliação dos pontos fortes e fracos, possibilitando assim, ajustes e melhorias em projetos subseqüentes. Foi possível também monitorar o trabalho dos integrantes da fábrica através da utilização de *timesheets*, onde dados-históricos foram coletados, permitindo assim, o conhecimento das horas consumidas por fase;

v. Estimativas*:* o multiplicador de Ponto de Caso de Uso Ajustado (PCUA) para Homem/Hora (HH) utilizado durante o piloto foi 20, como sugerido pelo criador do método em [10], para empresas que não possuem dados de projetos anteriores. Dessa forma, resultou-se em uma estimativa de 445HH, porém, o total de tempo realmente gasto foi 350 horas.

Em virtude destes fatores, e pela própria execução do piloto, a fábrica sofreu alguns ajustes:

***i. Realocação de Perfis**:* foi realizado a realocação de três participantes em relação aos perfis funcionais, para melhor adequação aos futuros projetos;

***ii. Ajustes de Métricas**:* baseando-se nos dados históricos coletados, o multiplicador de Ponto de Caso de Uso Ajustado (PCUA) para Homem/Hora (HH) foi modificado de 20 para 18;

***iii. Instrumentação**:* a ferramenta CASE utilizada foi substituída em razão das limitações encontradas.

2.3 Estudo de Caso: Projeto Real

2.3.1 Contexto

Após a estrutura proposta ter sido avaliada com a utilização de um projeto piloto, que possibilitou a coleta de métricas e resultou em ajustes para melhorar a organização geral da fábrica, um novo projeto foi desenvolvido. Visto que os participantes já tinham dados históricos e dispunham de um processo refinado, resolveu-se implantar uma maior rigorosidade, firmando um compromisso entre a prestadora de serviços (fábrica em questão) e o cliente, através da adoção de um Acordo de Nível de Serviço (SLA) [11]. Este define em maiores detalhes, e, com clareza, o relacionamento entre as partes, deixando explícito as obrigações de cada um nas interações entre as empresas e quais as penalidades que devem ser aplicadas caso o acordado não seja cumprido.

2.3.2 Execução

O projeto desenvolvido foi o do software SI Alocação Plus, sendo que o prazo acordado para entrega foi de seis (6) semanas. O objetivo do sistema foi o de solucionar o problema de alocação de salas (aulas, laboratórios, auditórios), professores, alunos e disciplinas, de forma automática e contemplando a eliminação de conflitos, utilizando técnicas de alocação de recursos. Este problema é conhecido na literatura como Timetabling e foi resolvido utilizando-se algoritmos genéticos [3].

O Acordo de Nível de Serviço estabelecido considerava como pontos de cobrança os seguintes indicadores:

- **I1.** Quantidade de problemas ocorridos no projeto e que não estavam registrados como riscos e/ou não tinham ações de contingência previstas;
- **I2.** Soma dos dias de atraso/adiantamento nas entregas de artefatos;
- **I3.** Quantidade de relatórios semanais de status do projeto que não foram enviados, ou que não estavam completos em relação a eventos relevantes.

Além disso, existiu uma tolerância em relação ao valor dos contadores, desde que eles não ultrapassassem a meta, que consistia na manutenção do saldo (bônus + penalidades) zerado, aplicando-se a seguinte fórmula para avaliação: Saldo = (I1-1) + (I2-3) + (I3). Como foi a primeira vez que este tipo de acordo foi aplicado, combinou-se que os valores dos indicadores e as suas compensações seriam registrados para efeito de aprendizado, mas não aplicados.

As atividades relacionadas à garantia de acurácia dos artefatos foram efetuadas com maior rigor e constituíam-se na validação, por duas instâncias, dos documentos gerados: líder de equipe e analistas de qualidade; dessa forma, tinha-se uma avaliação mais criteriosa e, conseqüentemente, o produto final disponibilizado apresentava uma maior qualidade.

A Figura 3 apresenta os dados coletados, como a quantidade de horas utilizadas por fase durante a execução do projeto.

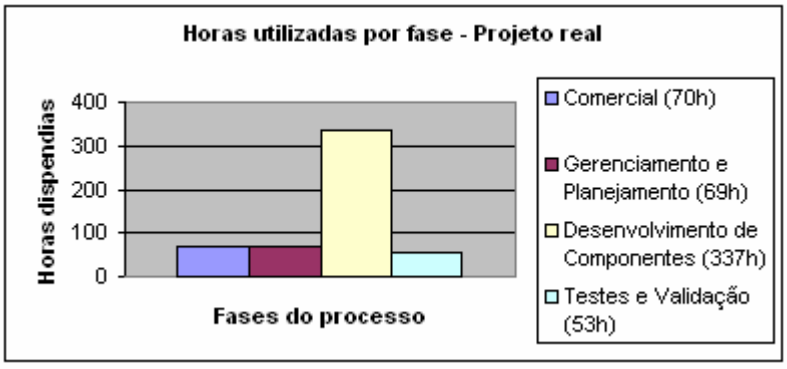


Figura 3: Horas utilizadas por fase no projeto real

As demais etapas foram similares às desenvolvidas durante o projeto piloto, com a vantagem de que a execução deste foi bastante favorecida devido à experiência adquirida durante o primeiro, resultando numa uniformização maior do número de horas trabalhadas e evitando a sobrecarga dos membros da equipe de desenvolvimento.

2.3.3 Análise

Ao usufruir o aprendizado adquirido com a realização do projeto piloto, algumas atividades foram realizadas de melhor modo, destacando-se as seguintes:

i. Estimativas*:* com o multiplicador ajustado, as horas estimadas foram 565 e o tempo realmente gasto foi 541 horas, ou seja, o esforço estimado mostrou-se bem mais próximo ao real do que anteriormente, isto demonstra um bom planejamento e gera segurança para estimar valores futuros;

ii. Realocação de perfis*:* o novo cenário mostrou-se vantajoso, visto que as atribuições estavam mais adequadas às habilidades dos participantes, resultando em uma maior eficiência e produtividade. A atividade de liderança nas equipes foi realizada de forma mais efetiva, permitindo um maior controle sobre a evolução de cada grupo específico;

iii. Maturidade do processo*:* após refinamento e melhor conhecimento dos membros envolvidos acerca do processo utilizado, foi possível segui-lo de maneira mais direta.

Um ajuste importante ainda se faz necessário:

i. Evolução do SLA*:* um acordo de nível de serviço deve estar bem entendido e aculturado para todas as partes antes de ser implantado, para que ele não se transforme em um problema e cumpra o seu papel. Dessa forma, recomenda-se evoluir de forma não brusca e tentar seguir a evolução do relacionamento entre os envolvidos. As alterações devem ser decididas entre as partes e apenas entram em vigor para os novos projetos que se iniciarem após essas decisões consensuais.

3. Trabalhos Relacionados

Alguns trabalhos envolvendo fábricas de software existem na literatura. Nesta seção, apresentaremos alguns deles, enfatizando suas diferenças e similaridades em relação ao presente trabalho.

Em [14], os autores apresentam uma vasta coleção de dados de entrevistas, questionários e métricas de software, a fim de descrever como criar e manter uma organização efetiva, centrada em processos de desenvolvimento de software.

Fernström [7] descreve o projeto *Eureka Software Factory* (ESF), que tem por objetivo permitir a criação de um mercado para produtos CASE, que possam ser configurados para aplicações específicas, ajudando os construtores de fábricas a integrar os produtos e informações utilizadas ao longo do processo.

A ESF representa um esforço conjunto de treze entidades parceiras de cinco países europeus, entre as quais encontram-se instituições de pesquisa, fabricantes de computadores, desenvolvedores de ferramentas CASE e sistemas.

A grande diferença deste trabalho em relação aos demais, está no relato das experiências com uma fábrica de software, desde a sua fase de definição até as lições aprendidas durante todo o processo.

4. Conclusões e Trabalhos Futuros

Neste artigo, apresentamos o processo de criação de uma fábrica de software onde foram discutidas, detalhadamente, as etapas de definição e implantação, ressaltando as lições aprendidas durante a execução dos projetos experimentais.

A execução dos projetos experimentais foi de grande importância para a apreciação do processo utilizado pela fábrica, permitindo a realização de ajustes necessários, como realocação de participantes a perfis apropriados. O êxito conseguido durante a instanciação da fábrica mostrou a viabilidade do processo adotado, além de servir como fator de motivação para realização de outros projetos.

A principal contribuição deste trabalho está numa detalhada descrição de como criar uma fábrica de software sem muitos recursos físicos e financeiros, aspecto este, não mencionado com muita freqüência pela literatura.

Como trabalhos futuros, pretende-se realizar projetos de grande porte, a fim de realizar alguns ajustes e refinar ainda mais os processos anteriormente definidos, visando sempre a melhoria contínua.

Agradecimentos

Os autores desejam agradecer o apoio de todos os integrantes envolvidos na construção e implantação da Fábrica de Software apresentada neste trabalho, a qual foi resultante de um projeto requerido pela disciplina Engenharia

de Software oferecida aos alunos da pós-graduação do Centro de Informática na Universidade Federal de Pernambuco.

Referências

[1] Almeida, E, S., Bianchini, C, P., Prado, A, F., Trevelin, L, C., *IPM: An Incremental Process Model for Distributed Component-Based Software Development*, In The 5th International Conference On Enterprise Information Systems (ICEIS), Angers - France. ACM Press, 2003.

[2] Almeida, E, S.; Lucrédio, D; Bianchini, C, P.; Prado, A, F.; Trevelin, L, C. Ferramenta MVCase - Uma Ferramenta Integradora de Tecnologias para o Desenvolvimento de Componentes Distribuídos. In: Simpósio Brasileiro de Engenharia de Software (SBES) - Sessão de Ferramentas, 2002, Gramado/RS - Brazil.

[3] Caldeira, P., Agostinho, R., *School Timetabling Using Genetic Search*, Practice and Theory of Automated Timetabling, Toronto, 1997.

[4] Cleland, D., *A Guide to the Project Management Body of Knowledge*, Project Management Institute, 2000.

[5] Coulter, N., Dammann, J, *Current practices, culture changes, and software engineering education*. Computer Science Education, 5, 2, 211 – 227, 1994.

[6] Dawson R.J., Newsham, R.W. and Kerridge, R.S, *Introducing new software engineering graduates to the 'real world' at the GPT company*. Software Engineering Journal, Vol. 07, No. 03, May, 1992.

[7] Fernström, C., Närfelt, K., Ohlsson, L., *Software Factory Principles, Architecture, and Experiments*, IEEE Software, Vol. 09, No. 01, January, 1992.

[8] Gibbs N., *The SEI education program: The Challenge of Teaching Future Software Engineers*, Communications of the ACM, Vol. 32, No. 05, May, 1989.

[9] Jacobson, I., et. al., *The Unified Software Development Process*, Addison-Wesley, 2001.

[10] Karner, G., *Resource Estimation for Objectory Projects*, Copyright Rational Software, 1993.

[11] Karten, N. *Establishing Service Level Agreements*, 2001. In http://www.nkarten.com/, último acesso em maio de 2004.

[12] Meneses, J. B., Moura, H. P., *Inspector: Um Processo de Avaliação de Progresso para Projetos de Software*, Simpósio Brasileiro de Engenharia de Software, Rio de Janeiro, 2001.

[13] Shaw M., *Prospects for an Engineering Discipline of Software*. IEEE Software, Vol. 07, No. 06, November, 1990.

[14] Siy, H, P., Mockus, A., Herbsleb, J, D., Krishnan, M., Tucker, G, T., *Making The Software Factory Work: Lessons from a decade of experience*, In the Seventh International Symposium on Software Metrics, London, England, April 2001.

[15] Wasserman A.I. *Toward a Discipline of Software Engineering*. IEEE Software, Vol. 13, No. 06, November 1996.

Uma Proposta para o Mapeamento entre a API DOM e o Padrão MOF

Hélio Lopes dos Santos
Universidade Federal de Pernambuco(UFPE), Centro de Informática(CIn),
Recife, Brasil, 50.732-970
hls@cin.ufpe.br

Maísa Soares dos Santos
Universidade Estadual do Sudoeste da Bahia (UESB), Departamento de Ciências Exatas (DCE),
Vitória da Conquista, Brasil, 45.000-000
maisa@uesb.br

Roberto Souto Maior de Barros
Universidade Federal de Pernambuco(UFPE), Centro de Informática (CIn),
Recife, Brasil, 50.732-970
Roberto@cin.ufpe.br

Abstract

This paper presents a metamodel aimed at mapping DOM interfaces into MOF structures such as packages, classes, associations, etc. The DOM API makes XML data available to programs in the form of objects. The MOF standard defines an abstract language and a framework aimed at specifying, implementing and managing platform independent metamodels. MOF tools will be able to manage all kinds of XML documents using the proposed metamodel.
Keywords: XML, DOM, MOF, Metamodel.

Resumo

Este artigo apresenta um metamodelo que faz o mapeamento entre as interfaces do DOM e os elementos do MOF como pacote, classe, associações, etc. A API DOM disponibiliza dados para as aplicações XML em forma de objetos. O padrão MOF define uma linguagem abstrata e um framework para especificação, construção e gerenciamento de metamodelos independentes de tecnologia de implementação. As ferramentas que implementam o padrão MOF poderão gerenciar qualquer tipo de documento XML utilizando o metamodelo proposto.
Palavras chave: XML, DOM, MOF, Metamodelo.

1. INTRODUÇÃO

Nos últimos anos, com o crescimento dos sistemas de informação, metadados tornaram-se peças chave no gerenciamento de todo o ciclo de vida desses sistemas. Muitos esforços recentes, tanto das áreas acadêmicas quanto da indústria, estão sendo concentrados em pesquisas relacionadas a metadados [8,1,18]. Um dos produtos dessas pesquisas é o MOF (*Meta Object Facility*) [10]. Este padrão foi projetado pela OMG (*Object Management Group*) especificamente para dar suporte à modelagem, gerenciamento e intercâmbio de metadados e está voltado ao contexto de sistemas de informação. O MOF possui como uma de suas características principais permitir interoperabilidade entre ferramentas de software que produzem e compartilham metadados.

A partir dessas pesquisas, verificou-se também que o padrão XML é adequado para ser utilizado como formato de descrição, armazenamento e intercâmbio de metadados, pois XML representa dados semi-estruturados, flexível, voltado para o contexto web e suportado por uma variedade de ferramentas [1]. Para dar suporte às suas funções, XML incorpora alguns padrões, como o DOM (*Document Object Model*) [4] utilizado para acessar e manipular documentos XML.

Este artigo propõe o projeto e a implementação de um metamodelo MOF para o DOM. Um metamodelo define um modelo para construir modelos. Ele é um conjunto de metadados inter-relacionados utilizados para definir modelos [8]. O metamodelo MOF proposto permite que qualquer documento XML seja gerenciado por uma ferramenta que implemente o padrão MOF. Atualmente, o MOF permite importar documentos XML apenas no formato XMI (*XML Metadata Interchange*) [7], que é o formato padrão para intercâmbio de metadados do MOF. Com o metamodelo proposto os usuários poderão gravar documentos XML com qualquer conjunto de tags, não necessariamente sendo XMI.

Uma das vantagens desta proposta é um conjunto de interfaces comuns para acessar e gerenciar os metadados descritos no metamodelo MOF proposto. Estas interfaces são equivalentes às interfaces do DOM e permitem gerenciar documentos XML em repositórios MOF. Atualmente, o MOF permite gerar interfaces CORBA, através do mapeamento IDL -> CORBA [10] e Java, através do JMI (*Java Metadata Interface*) [7].

Uma outra vantagem é a utilização do padrão XMI como uma alternativa a mais para intercâmbio dos metadados. O XMI é um padrão suportado por várias ferramentas nas mais diversas áreas como desenvolvimento de sistemas, por exemplo, ferramenta *Netbeans*[1]*; e*specificação e modelagem de sistemas, por exemplo, Rational Rose[2] e ARGO UML[3]; ferramentas de *Data warehouse,*como o DB2 Warehouse Manager[4] e Oracle Warehouse Builder[5]; repositórios de metadados como, por exemplo, Universal Repository (UREP)[6], MDR (*Metadata Repository*) [17] e dMOF [3].

Nas próximas seções são apresentados os trabalhos relacionados, o padrão DOM e MOF. Logo após, é apresentada a arquitetura e o padrão de mapeamento para interfaces Java e CORBA do MOF. É apresentado também o metamodelo MOF proposto para o padrão DOM, as interfaces geradas a partir deste metamodelo e um estudo de caso utilizando tais interfaces. E, na última seção, são apresentados as conclusões e os trabalhos futuros.

2. TRABALHOS RELACIONADOS

Outros metamodelos baseados no padrão MOF são o UML (*Unified Modeling Language*) [11] e o CWM (*Common Warehouse Metamodel*) [14].

A UML é uma linguagem para especificação, visualização, construção e documentação de artefatos de software. Ela provê uma linguagem de modelagem orientada a objetos, independente de plataforma, com conceitos bem definidos como classes, objetos, uses cases, associação, generalização, e diagramas gráficos.

O CWM é o padrão da OMG para integração de ferramentas de *Data Warehousing*. Essa integração se dá pelo compartilhamento de metadados. CWM define um conjunto de metamodelos para as diversas subáreas de *Data Warehousing*. CWM utiliza UML para definir os diversos metamodelos, chamados também de pacotes. Cada pacote possui um metamodelo sobre um domínio específico, mas existem dependências entre os mesmos.

O artigo [15] discute a integração entre padrões de metadados RDF (*Resource Description Framework*) e RDF Schema do W3C (*Word Wide Web Consortiun*) e o MOF da OMG. O artigo [16] apresenta um metamodelo para gerenciar metadados em XML e DTD, porém, tal metamodelo não foi modelado a partir da API DOM, além do que, o artigo discute mais o processo de desenvolvimento da aplicação de metadados, ressaltando as vantagens de se utilizar XML. Enquanto que este artigo foca mais no processo de mapeamento entre a API DOM e as interfaces geradas pelo repositório MOF.

Uma outra iniciativa importante da empresa *Sun Microsystem* é um metamodelo para a linguagem Java [2], que propõe uma maneira de gerenciar código de programas Java em repositórios MOF.

---

[1] http://www.netbeans.org/
[2] http://www.rational.com/
[3] http://argouml.tigris.org/
[4] http://www-3.ibm.com/software/data/db2/datawarehouse/
[5] http://otn.oracle.com/products/warehouse/content.html
[6] http://www.unisys.com/

3. DOM – DOCUMENT OBJECT MODEL

A necessidade de prover acesso padronizado às informações em documentos XML levou a W3C (*World Wide Web Consortium*) a desenvolver o DOM. De acordo com a definição deste consórcio, DOM é uma API (*Application Programming Interface*) para documentos XML e HTML, que define a estrutura lógica de documentos e a forma como esses documentos são acessados e manipulados [4].

O objetivo principal de DOM é fornecer uma interface de programação padrão que possa ser usada em uma grande variedade de ambientes e aplicações. A especificação DOM oferece apenas as definições de interfaces para as bibliotecas de DOM, e estas interfaces são independentes de plataforma e de linguagem [4].

DOM é uma estrutura de dados abstrata que representa os documentos XML como uma árvore de nós. O DOM compreende um conjunto de interfaces que oferece os serviços para gerenciar documentos XML. Um *parser*, ou seja, uma ferramenta que implementa este conjunto de interfaces, lê todo o documento e constrói uma árvore de objetos na memória. A partir dessa árvore é possível navegar por sua estrutura, adicionar, modificar, e remover elementos do documento.

Um documento XML é uma árvore composta de nós de vários tipos, como: elemento, atributo, texto, comentário, entidade, instrução de processamento, etc. Para cada um desses tipos, DOM define uma interface. Todos os nós na árvore DOM são uma instância da interface Node. Através dela é possível navegar e manipular a árvore. Node fornece métodos para obter propriedades como nome, valor, tipo, lista de filhos e atributos de um nó, bem como para inserir, remover e substituir nós.

Um elemento do documento XML é representado pela interface Element. A maioria dos métodos de Element manipula atributos. Esta interface declara funções que possibilitam verificar a existência e obter os atributos de um elemento, e adicionar e remover atributos.

Um atributo é representado pela interface Attr. Os métodos definidos em Attr permitem obter o nome completo (*namespace* URI e nome local) do atributo, verificar se ele é especificado no documento ou não, obter o elemento ao qual o atributo pertence, além de obter e alterar o valor do atributo.

O W3C mantém atualmente três especificações para DOM: DOM 1 [4], DOM 2 [5] e DOM 3 [6]. O nível 2 de DOM acrescenta algumas funcionalidades ao DOM1, entre elas suporte a *namespace*, módulos de eventos, de folha de estilo, de atravessamento e intervalo de documentos e de visões. O DOM 3 focaliza o carregamento, salvamento, além de suporte para validação de documentos, entre outros.

4. MOF – META OBJECT FACILITY

O MOF define uma linguagem abstrata e um *framework* para especificação, construção e gerenciamento de metamodelos independentes de tecnologia de implementação. Alguns exemplos incluem o metamodelo UML, CWM e o próprio MOF. O MOF possui ainda um conjunto de regras para implementação de repositórios, que manipulam metadados descritos pelos metamodelos. Essas regras definem um padrão de mapeamento entre os metamodelos MOF e um conjunto de APIs para gerenciamento de metadados, instâncias do metamodelo. Por exemplo, o mapeamento *MOF -> IDL* (*Interface Definition Language*) é aplicado aos metamodelos MOF (UML, CWM) para produzir uma API CORBA que gerenciem os metadados, instâncias desses metamodelos. O mapeamento *MOF -> Java*, através do padrão JMI (*Java Metadata Interface*), define as mesmas regras de mapeamento para API Java. Este trabalho utiliza JMI.

A especificação MOF compreende o seguinte:

- Uma definição formal para o meta-metamodelo MOF, ou seja, uma linguagem abstrata para a definição dos metamodelos.
- As regras para o mapeamento dos metamodelos MOF para uma tecnologia de implementação, por exemplo, CORBA ou Java.
- Padrão XMI para intercâmbio, em XML, dos metadados e metamodelos entre as ferramentas. XMI define um conjunto de regras que mapeiam os metamodelos MOF e os metadados em documentos XML.

4.1 Arquitetura de Metadados da OMG

O MOF é um *framework* extensível, ou seja, novos padrões de metadados podem ser adicionados ao mesmo. Para isto, conta com uma arquitetura em quatro camadas, também chamada de *OMG Meta Data Architecture* [10, 13], mostrada na Tabela 1.

Tabela 1 – A arquitetura de metadados da OMG

Nível MOF	Termos Utilizados	Exemplos
M3	Meta-Metamodelo	Modelo MOF
M2	Metamodelo, Meta-Metadados	Metamodelo UML e CWM
M1	Modelos, Metadados	Modelos UMLs – diagramas de classes, Esquemas Relacionais instâncias do metamodelo CWM da camada M2
M0	Dados, Objetos	Dados

A instância de uma camada é sempre modelada por uma instância de uma camada imediatamente superior. Desta forma, a camada M0 onde estão os dados são modelados por modelos UML, como diagramas de classes que estão na camada M1. A camada M1 por sua vez é modelada pelo metamodelo UML, camada M2, que utiliza os construtores básicos como classes, relacionamentos, entre outros. Este metamodelo é uma instância do modelo MOF, que é chamado também de meta-metamodelo. Um outro exemplo, um modelo na camada M1 que esteja no padrão CWM, é uma instância do metamodelo CWM na camada M2. A extensibilidade se dá pelo fato de, em cada camada, podermos adicionar classes que são instâncias de outra classe de uma camada imediatamente superior.

4.2 O Modelo MOF

Como foi visto na seção anterior, o metametamodelo MOF, também chamado de modelo MOF, está situado na quarta camada (M3) da arquitetura de metadados da OMG. Esta seção descreve os seus principais aspectos.

O MOF é descrito em termos dos seus próprios conceitos, ou seja, ele é auto descritivo. Isto significa que o modelo MOF é o seu próprio metamodelo. A especificação MOF o descreve de forma narrativa e através do uso de notação UML, tabelas e expressões OCL (*Object Constraint Language*) [11]. UML é utilizada apenas por possuir uma representação gráfica dos modelos, o que facilita a leitura por parte dos leitores. Ela não define a semântica do modelo MOF, que está completamente definida na especificação MOF e não depende da semântica de qualquer outro modelo. O MOF não especifica qual linguagem deve ser utilizada para definir as restrições dos seus metamodelos, porém ele utiliza como linguagem padrão a OCL.

O MOF é um modelo orientado a objetos e possui um conjunto de elementos de modelagem que são utilizados na construção dos metamodelos, incluindo regras para o seu uso. São exemplos destes elementos, classes, associações, pacotes, tipos de dados, constantes, entre outros.

As interfaces geradas a partir do mapeamento *MOF -> Plataforma_especifica* compartilham um conjunto comum de quatro tipos de meta objetos: *instance*, *class proxy, association* e *package*.

***Package* –** Uma instância da classe MOF *Package*. Um meta objeto pacote representa um repositório *container* de outros tipos de meta objetos. O pacote mais externo *(outer most package)* representa o meta objeto raiz do modelo de meta objetos.

Class proxy – Cada classe do nível M2 possui uma classe *proxy* correspondente. Existe um objeto *proxy* para cada classe do nível M2. Este tipo de meta objeto serve para produzir meta objetos do tipo *instance*.

Instance – As instâncias das classes do nível M2, ou seja, dos metamodelos, são representados pelos meta objetos do tipo *instance*. Um meta objeto *instance* manipula os estados dos atributos das classes do nível M2.

***Association* –** Esses objetos manipulam as coleções de *links* correspondentes às associações do nível M2. São objetos estáticos e possuem como *containers* meta objetos do tipo *package*. As interfaces dos objetos do tipo *Association* disponibilizam operações para consultar um *link* no conjunto de *links; o*perações para adicionar, modificar e remover *links* a partir do conjunto; e operações que retornam todo o conjunto de *links*.

4.3 O Mapeamento MOF -> CORBA IDL/Java

Atualmente, faz parte da própria especificação MOF o mapeamento MOF -> CORBA IDL. A comunidade Java definiu o mapeamento MOF->Java e o chamou de JMI (*Java Metadata Interface*) [7]. As regras gerais de mapeamento são praticamente as mesmas para qualquer plataforma.

O resultado deste mapeamento é um conjunto de interfaces que permite ao usuário criar, alterar e consultar metadados, instâncias dos metamodelos MOF. Por exemplo, se as interfaces forem geradas a partir do mapeamento MOF->CORBA IDL, o usuário pode utilizar clientes CORBA para acessar as interfaces. Se for JMI, o usuário poderá utilizar clientes Java.

Esta seção descreve o padrão de herança para as interfaces mapeadas a partir dos metamodelos MOF. A Figura 1 apresenta um exemplo de metamodelo MOF expresso em UML que consiste de dois pacotes P1 e P2. O primeiro pacote P1 contém as classes C1 e C2, onde C2 é subclasse de C1 e uma associação A que conecta C1 e C2. O segundo pacote P2 é definido como subpacote de P1.

A Figura 2 apresenta o diagrama UML que mostra gráfico de herança gerado a partir do mapeamento MOF para Java. A raiz do gráfico é um grupo de interfaces que fazem parte do pacote reflexivo do MOF. Toda interface gerada a partir dos metamodelos herda direta ou indiretamente das interfaces reflexivas.

As regras de mapeamento dizem que para cada pacote e para cada associação do metamodelo são criadas uma interface *package* e uma *association,* respectivamente. Para cada classe do metamodelo são criadas uma interface *proxy* e uma interface *instance*. O padrão de herança segue as seguintes regras:

- Um meta objeto instance, que não possui supertipo, herda de RefObject; todos os outros meta objetos instances estendem seus supertipos.
- Um meta objeto package, que não possui supertipo, herda de RefPackage; todos os outros meta objetos package estendem seus supertipos.
- Todos os meta objetos class proxy herdam de RefClass.
- Todos os meta objetos association herdam de RefAssociation;

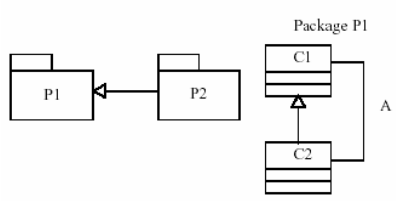


Figura 1 - Exemplo de um metamodelo MOF

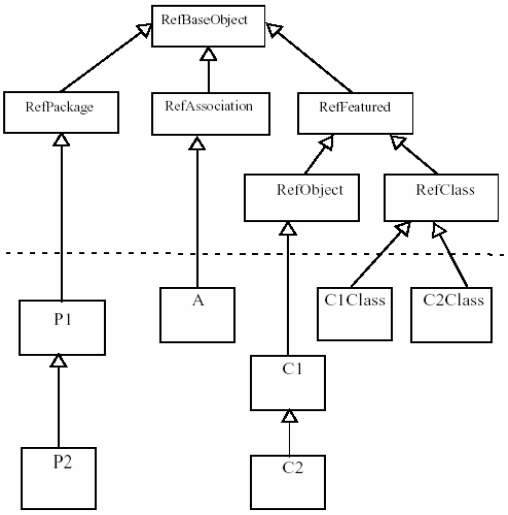


Figura 2 - Exemplo do metamodelo MOF mapeado para interfaces Java

Para o exemplo da Figura 2, foram geradas duas interfaces *package* referentes aos pacotes P1 e P2. O pacote P1, que não possuía supertipo, herdou de *RefPackage*, enquanto P2, que possuía, herdou de P1. Foi gerada também a interface *A* para a associação entre as classes C1 e C2 do metamodelo. Para cada classe do metamodelo foram geradas duas classes: uma para os meta objetos *instances* e outra para as *class proxy*. As interfaces *C1Class* e *C2Class* representam as interfaces *proxy* geradas, respectivamente, a partir das classes C1 e C2 do metamodelo, e herdam diretamente de *RefClass*. As interfaces *C1* e *C2* representam as interfaces para os meta objetos instances. Apenas a interface *C1*, cuja classe do metamodelo não possuía supertipo, herdou de *RefObject*.

5. O METAMODELO MOF PROPOSTO PARA O PADRÃO DOM

Diversas ferramentas suportam o MOF, como por exemplo, o MDR (*Metadata Repository*) [17], dMOF [3] e UREP (*Universal Repostiory*). Algumas fazem o mapeamento para CORBA (dMOF) e outras fazem para JMI (MDR). Algumas armazenam os objetos em SGBD (Sistema Gerenciador de Banco de Dados) oferecendo maior escalabilidade e robustez para as aplicações de metadados. Esta seção apresenta o metamodelo DOM que poderá ser utilizado por estas ferramentas para dar suporte ao gerenciamento de metadados descritos em XML.

O metamodelo proposto é um mapeamento direto entre as interfaces do DOM e os elementos do MOF como pacotes, associação, classes, etc. Por exemplo, a classe *DOMNode* é a mais genérica do metamodelo e representa a interface *Node* do DOM. Ela possui os atributos *nodename* e *nsURI* que representam, respectivamente, o nome do *nó* e o seu *namespace*. Muitos métodos da interface *Node* não foram mapeados para a classe *DOMNode*. Por exemplo, os métodos para manipular os *nós* filhos como inserir, remover, alterar, recuperar, etc. Esses métodos são encontrados na classe que representa a associação *DOMContains* do metamodelo.

Nem todas as subclasses usam efetivamente todos os atributos da classe *DOMNode*. Por exemplo, o nó comentário (*DOMComment*) não possui *namespace* e o seu valor é representado pelo próprio nome do *nó*.

A associação *DOMContains* permite a composição de *nós* XML. Alguns *nós* não podem conter outros, por exemplo, um nó texto ou atributo não pode conter outros nós. Para que o modelo fique consistente foi necessário adicionar *constraints* OCL ao metamodelo que restringem algumas composições possíveis segundo a própria especificação DOM.

A classe *XMLDocumentType* representa a interface *DocumentType* de DOM. Os textos são representados pela classe *DOMText*. Os elementos XML são representados pela classe *DOMElement* que representa a interface *Element* de DOM. A classe *DOMAttr* representa a iterface *Attr* de DOM que manipula atributos de XML. As interfaces *Entity*, *EntityReference, ProcessingInstrucion* são mapeadas, respectivamente para as classes *DOMEntity*, *DOMEntityReference, DOMProcessingInstrucion.* A classe DOMNodeValue não possui nenhuma interface correspondente na API DOM, porém ela foi acrescentada para oferecer um atributo *nodevalue* que é comum a algumas classes do metamodelo como entidades, atributos e instruções de processamento. Um documento XML é representado pela interface *Document* em DOM e pela classe *DOMDocument* no metamodelo MOF. Em DOM, os métodos para a criação dos *nós* como atributos e elementos fazem parte da interface *Document*. Em MOF, estes métodos farão parte das classes *proxys*.

A Figura 3 apresenta o metamodelo DOM em notação UML. Todos os componentes de um metamodelo MOF (classes, atributos, associações) devem estar dentro de um ou mais pacotes. Para o metamodelo DOM foi criado o

pacote *DOMMetamodel*. As classes que estão em cinza são abstratas, ou seja, elas não podem instanciar objetos diretamente, apenas suas subclasses.

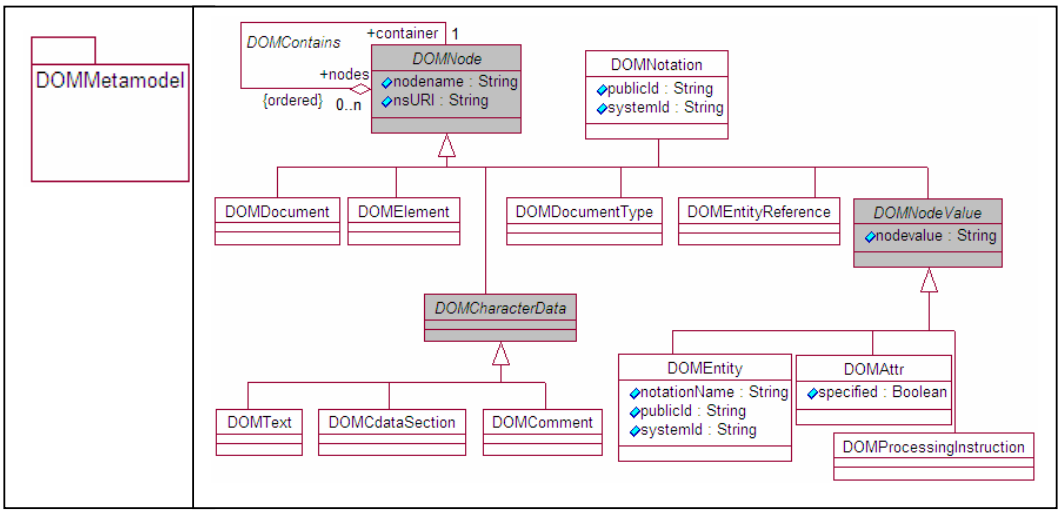


Figura 3 – O metamodelo DOM proposto – DOMMetamodel

Algumas informações não puderam ser apresentadas no metamodelo, como por exemplo, as restrições (*constraints)*. Estas foram acrescentadas diretamente ao documento XMI referente ao metamodelo DOM e limitam alguns tipos de agregações entre os *nós* XML. Como explicado na Seção 4.2, o MOF utiliza OCL como linguagem para a definição de restrições. A Figura 4 apresenta um exemplo de *constraint* descrita em XMI que diz que um documento XML deve possuir exatamente um elemento raiz. A ferramenta MDR ainda não suporta *constraints*, porém qualquer outra ferramenta que implemente a especificação MOF e suporte *constraints* poderia utilizá-las.

```
- <Model:Constraint annotation="" evaluationPolicy="deferred" language="OCL"
    xmi.id="C0">
    <Model.ModelElement.name
      xml:space="preserve">DOMDocumentContemUmDOMElement</Model.ModelElement.name>
    <Model.Constraint.expression xml:space="preserve">context document :
      DOMDocument inv C0:document.nodes->select(obj:DOMNode |
      (obj.oclIsTypeOf(DOMElement)))->size=1</Model.Constraint.expression>
  - <Model:Constraint.constrainedElements>
      <Model:ModelElement xmi.idref="a3D2A1A46009D" />
    </Model:Constraint.constrainedElements>
  </Model:Constraint>
```

Figura 4 – Exemplo de uma constraint OCL

O atributo *language* refere-se à linguagem em que está escrita a *constraint* enquanto que *evaluationPolicy* pode possuir os valores *immediate* ou *deferred* e se refere ao momento para a verificação de violação da *constraint*. Além dos atributos, existem também os subelementos *Model:ModelElement.name* e *Model:Constraint.expression* que armazenam, respectivamente, o nome e a própria *constraint*, e *Model:Constraint.contrainedElements* que diz a quais elementos do metamodelo tal *constraint* se aplica.

5.1 O As Interfaces Geradas a partir do Metamodelo DOM

Os usuários podem gravar metadados no repositório de duas maneiras. A primeira é importando os metadados descritos em documentos XMI. A segunda é através das interfaces geradas a partir do metamodelo. Essas interfaces permitem ao usuário criar, alterar e acessar as instâncias do metamodelo, que são os metadados, utilizando programas Java.

Foi gerado o documento XMI referente ao metamodelo construído e, após, foi importado para o repositório MOF, implementado pela ferramenta MDR.

Como foi visto na Seção 4.4, a especificação MOF define um conjunto de regras para a geração de um conjunto de interfaces a partir de um metamodelo específico. Uma das regras diz que para cada classe do metamodelo, duas interfaces são geradas: a primeira representa as instâncias da classe, os meta objetos *instance*, e a segunda representa uma classe *proxy*, os meta objetos *class proxy*.

A Listagem 1 apresenta a interface *DOMNode* gerada a partir do metamodelo DOM. Essa interface oferece métodos para acessar e manipular o estado dos atributos e as referências da classe do metamodelo. A Listagem 2 apresenta duas interfaces *proxys*: uma para *DOMNode* e outra para *DOMAttr*. As interfaces *proxys* possuem operações que

criam instâncias de suas classes correspondentes, isto se a classe não for abstrata. A classe *DOMNode* do metamodelo DOM é abstrata, ela não instancia diretamente objetos, por isso a interface *DOMNodeClass* não possui operação para a criação de objetos do tipo *DOMNode*.

Listagem 1 - Interface Domnode do metamodelo DOM

```
public interface Domnode extends javax.jmi.reflect.RefObject {
    public java.lang.String getNodename();
    public void setNodename(java.lang.String newValue);
    public java.lang.String getNsURI();
    public void setNsURI(java.lang.String newValue);
    public dommetamodel.Domnode getContainer();
    public void setContainer(dommetamodel.Domnode newValue);
    public java.util.List getNodes();
}
```

Listagem 2 – Interfaces DomnodeClass e DomattrClass do metamodelo DOM

```
public interface DomnodeClass extends javax.jmi.reflect.RefClass {
}
public interface DomattrClass extends javax.jmi.reflect.RefClass {
          public Domattr createDomattr();
          public Domattr createDomattr(java.lang.String nodename,
                                       java.lang.String nsURI,
                                       java.lang.String nodevalue,
                                       boolean specified);
}
```

Para cada associação do metamodelo é gerada uma interface com um conjunto de métodos para acessar, alterar, inserir as instâncias das associações. Essas instâncias representam ligações entre os objetos, instâncias das classes do metamodelo. O metamodelo DOM possui a associação *DOMContains* que diz que um *nó* do tipo *DOMNode* poderá conter outros *nós* do mesmo tipo. A Listagem 3 apresenta a interface *DOMContains*.

Listagem 3 – Interface Domcontains do metamodelo DOM

```
public interface Domcontains extends javax.jmi.reflect.RefAssociation {
       public boolean exists(dommetamodel.Domnode nodes,
                             dommetamodel.Domnode container);
       public java.util.List getNodes(dommetamodel.Domnode container);
       public dommetamodel.Domnode
                        getContainer(dommetamodel.Domnode nodes);
       public boolean add(dommetamodel.Domnode nodes,
                          dommetamodel.Domnode container);
       public boolean remove(dommetamodel.Domnode nodes,
            dommetamodel.Domnode container);
}
```

Para cada pacote do metamodelo é gerada uma interface que contém métodos para acessar todos os objetos *proxys* referentes a todas as classes e associações do metamodelo. A Listagem 4 apresenta a interface para o pacote *DOMMetamodelPackage*, único pacote do metamodelo DOM.

Listagem 4 - Interface DommetamodelPackage do metamodelo DOM

```
public interface DommetamodelPackage extends
                                    javax.jmi.reflect.RefPackage {
    public dommetamodel.DomcdataSectionClass getDomcdataSection();
    public dommetamodel.DomcommentClass getDomcomment();
    public dommetamodel.DomdocumentClass getDomdocument();
    public dommetamodel.DomdocumentTypeClass getDomdocumentType();
    public dommetamodel.DomtextClass getDomtext();
    //unica associação do metamodelo
    public dommetamodel.Domcontains getDomcontains();
 ... }
```

Após a geração das interfaces, o próximo passo será a compilação dessas interfaces para serem utilizadas. A próximas seções ilustram o uso do metamodelo através das interfaces geradas.

5.2 O Mapeamento DOM -> MOF e MOF -> DOM

Uma das vantagens deste trabalho é permitir que repositórios MOF possam importar e exportar não somente documentos XMI, mas qualquer tipo de documento XML. Isto é importante, pois os usuários podem possuir ferramentas que produzem metadados em diversos formatos de XML, não necessariamente sendo XMI. Desta forma, basta importar os mesmos para o repositório MOF. Para isto, é necessário construir utilitários que façam o mapeamento entre os metadados descritos em XML para o seu metamodelo MOF.

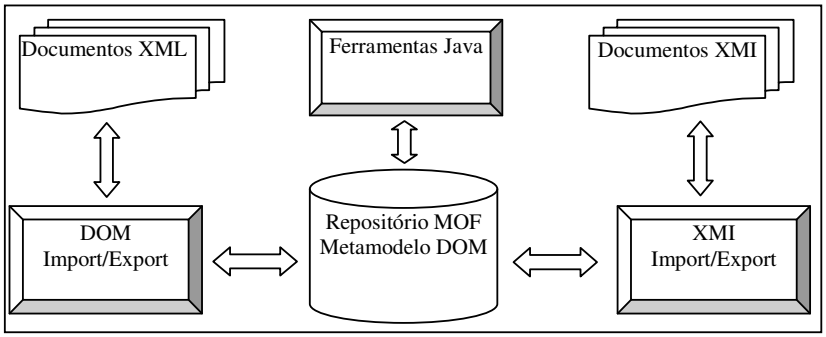


Figura 5 - Arquitetura dos Módulos Exportação e Importação de Documentos XML e XMI

A Figura 5 apresenta a arquitetura dos módulos que fazem a exportação e importação de documentos nos formatos XML e XMI para o repositório MOF. A importação e exportação de documentos XMI já é oferecida por qualquer ferramenta que implementa o MOF. Já a exportação e importação de documentos XML foi implementada através de um conjunto de métodos que transformam objetos do repositório MOF em objetos DOM e *vice-versa.* Além dos módulos de exportação e importação, o usuário poderá utilizar programas Java para acessar diretamente o repositório sem necessariamente usar as estruturas de DOM.

O processo de importação consiste em submeter o documento XML a um *parser* DOM, que processa o documento criando a estrutura de árvore de objetos na memória. Após, é necessário converter essa estrutura de objetos DOM em objetos do repositório MOF como instância do metamodelo DOM. O módulo *DOM Import* executa as seguintes tarefas:

1. Processar o documento XML, verificando se ele é bem formado, ou seja, se os dados do arquivo passado como parâmetro estão mesmo na sintaxe XML. Este passo criará a árvore de objetos DOM.
2. Iniciar o repositório MOF e pesquisar o repositório DOM, representado pelo metamodelo DOM. Esta tarefa consiste em achar uma instância da classe *MOFPackage* do metamodelo MOF cujo nome é *DOMMetamodel.*
3. Criar uma instância da classe *DOMDocument* que representa o novo documento a ser importado para o repositório.
4. Para cada objeto do tipo *Element* da estrutura DOM, criar uma instância da classe *DOMElement* no repositório MOF.
5. Para cada objeto do tipo *Attr* da estrutura DOM, criar uma instância da classe *DOMAttr* em MOF.
6. Similar aos passos 4 e 5, fazer os outros mapeamentos, como comentários, entidades, instrução de processamento, entre outros.
7. Além de criar os objetos, é necessário criar as ligações entre os mesmos através da classe *DOMContains,* única associação do metamodelo DOM.

Listagem 5- Parte do Código Java para Pesquisar um Documento XML no Repositório MOF

```
1 - if (pkg !=null){
2 -         javax.jmi.reflect.RefClass refclass =
                                  pkg.refClass("DOMDocument");
3 -         java.util.Collection c = refclass.refAllOfClass();
4 -         java.util.Iterator iter = c.iterator();
5 -         while(iter.hasNext()) {
6 -             Domdocument mofdoc = (Domdocument)iter.next();
7 -            if (doc.getNodename().equals(docname)){
8 -                Node node =
                   DOMDocumentFactory.mof2dom(xmldoc,mofdoc,pkg);
9 -                xmldoc.appendChild(node);
10-                saveXML(xmldoc); }
        }
 }
```

O processo de exportação consiste na geração de documentos XML a partir do repositório MOF. Para isto, é necessário inicializar o repositório MOF e pesquisar o sub-repositório DOM, ou seja, o pacote *DOMMetamodel.*

Após, consultar a instância da classe *DOMDocument,* dentro do pacote, que representa o documento XML a ser exportado. Esta instância é passada como argumento do método *mof2dom* da classe *DOMDocumentFactory*. Este método gerará a estrutura DOM a partir do objeto, instância da classe *DOMDocument*.
A Listagem 5 apresenta uma parte do código Java utilizado para pesquisar um objeto, instância da classe *DOMDocument* no repositório MOF. O método *refClass* (linha 2) da interface reflexiva *Refpackage,* retorna a interface *proxy* de uma determinada classe do metamodelo que é passada como parâmetro. Neste caso, a classe é *DOMDocument* e a interface retornada é *DOMDocumentClass*. O método *refAllOfClass* retorna uma lista de todos os objetos instância de uma determinada classe do metamodelo, no exemplo *DOMDocument*. A próxima tarefa é pesquisar, através do atributo *nodename*, o objeto na lista de objetos e passar para o método *mof2dom* (linha 8) que retorna um *node* XML contendo todo o documento XML.

6. ESTUDO DE CASO

Além de importar/exportar documentos XML, uma outra maneira de gerenciar documentos XML é utilizar diretamente as interfaces geradas a partir do metamodelo. Os usuários podem construir programas clientes que conectem ao repositório MOF, obtenham o pacote referente ao metamodelo e utilizem-no na construção de novos metadados que serão instâncias do metamodelo. A Listagem 6 apresenta uma parte de um código em Java para acessar o repositório DOM, representado pelo pacote *DOMMetamodel*.
A linha 1 do programa *MDRManager.getDefault().getDefaultRepository()* retorna o repositório padrão armazenado. Este repositório vai ser sempre o próprio metamodelo MOF. Após, é necessário pesquisar no repositório padrão o pacote *proxy* referente ao metamodelo DOM. O método *repository.getExtent("dommetamodel"),* na linha 2, procura um determinado pacote *proxy* dentro do repositório MOF. Ele recebe o nome do pacote como parâmetro e retorna um objeto do tipo *RefPackage,* que faz parte do pacote reflexivo de JMI. Se a pesquisa ao pacote desejado for bem sucedida, o próximo passo será utilizar o pacote para gerenciar metadados descritos em XML.

Listagem 6 – Código Java para criar documentos XML no repositório MOF

```
1 - MDRepository repository =
                        MDRManager.getDefault().getDefaultRepository();
2 - DommetamodelPackage pkg =
            (DommetamodelPackage)repository.getExtent("dommetamodel");
3 - if (pkg !=null)   {
4 -  repository.beginTrans(true);
5 -  Domdocument docroot =
     pkg.getDomdocument().createDomdocument("DATAWAREHOUSES.xml","");
6 -  DomprocessingInstruction   pi   =
  pkg.getDomprocessingInstruction().createDomprocessingInstruction(
                        "xml","","version=\"1.0\" encoding=\"UTF-8\"");
7 -  DomdocumentType dt=
                   pkg.getDomdocumentType().createDomdocumentType(
                            "\"C:\\Metadata\\dw.dtd\"");
8 -  Domelement elroot =
            pkg.getDomelement().createDomelement("DATAWAREHOUSES");
9 -  pkg.getDomcontains().add(pi,docroot);
10  -pkg.getDomcontains().add(dt,docroot);
11  -pkg.getDomcontains().add(elroot,docroot);
12  -pkg.getDomcontains().add(
       pkg.getDomcomment().createDomcomment("Metadados sobre os
                                    esquemas dos datawarehouses",""),
                                              docroot);
13  -repository.endTrans();
}
```

Para criar qualquer objeto no metamodelo é necessário obter a referência à interface *proxy* do objeto a ser criado. Por exemplo, para criar um novo documento XML é necessário *DomdocumentClass*, que é obtido pelo método *getDomdocument()* do pacote *DommetamodelPackage*. A interface *DomdocumentClass* possui o método para criar um novo documento XML, instância de *Domdocument.* Da mesma maneira, para se criar um objeto, instância de qualquer classe do metamodelo, é necessário obter a referência à sua interface *proxy*. Em DOM, os métodos para criar os objetos fazem parte da interface *Document,* em MOF, os métodos para a criação de objetos estão nas classes *proxys* referentes aos objetos.
A execução do código da Listagem 6 cria um objeto instrução de processamento, um objeto *DocType* e um objeto elemento, que é a raiz do documento XML. Além disso, é necessário fazer as associações entre os objetos. Isto é feito através da associação *DOMContains* do metamodelo. A Figura 6 apresenta o documento XML criado pela execução do código Java da Listagem 6.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE DATAWAREHOUSES SYSTEM "C:\metadata\dw.dtd">
<!--Metadados sobre os esquemas dos datawarehouses-->
<DATAWAREHOUSES></DATAWAREHOUSES>
```

Figura 6 - Documento XML criado através da execução do código da Listagem 6

7. Conclusões e Trabalhos Futuros

Este artigo apresentou o projeto e a implementação de um metamodelo MOF que visa representar objetos DOM em repositórios MOF. O padrão DOM oferece uma maneira das aplicações manipularem documentos XML em forma de objetos, dispostos em uma hierarquia, também chamada de árvore DOM. As ferramentas que implementam o MOF possuem a capacidade de importar metadados no formato XMI. Com o metamodelo DOM, essas ferramentas passam a importar e exportar qualquer tipo de documento XML. O padrão XML é independente de plataforma, semi-estruturado, suportado por uma variedade de ferramentas e adequado para representar metadados das mais diversas fontes.

Foi gerado um documento XMI a partir do metamodelo proposto. Qualquer repositório MOF poderá importar este documento e passará a ter suporte ao gerenciamento de metadados descritos pelo metamodelo. Além do metamodelo, os metadados que são instâncias desses metamodelo também poderão ser exportados e importados por qualquer repositório MOF.

Apresentamos também um estudo de caso ilustrando como as interfaces do metamodelo poderão ser utilizadas por outras ferramentas para o gerenciamento de metadados descritos pelo padrão modelado.

Um trabalho que poderia ser desenvolvido a partir deste seria a implementação do módulo de importação e exportação usando a API SAX (Simple API for XML) [9]. Esta interface é orientada a eventos. O *parser* que implementa SAX lê um documento e diz à aplicação quais os símbolos que encontra, à medida que os encontra. Uma das vantagens do uso desta API seria o uso de menos recursos computacionais para processar os documentos XML, pois esta não armazena o documento na memória.

References

[1].AHMED, Kal; AYERS, Danny; et al. "Professional XML Metadata". 2001. UK. Wrox Press Ltda.

[2].DEDIC, Svata; MATULA, Martin. "Metamodel for the Java language". Disponível em: http://java.netbeans.org/models/java/java-model.html. 2002.

[3].Distributed Systems Technology Centre - DSTC. "DMof - An OMG Meta Object Facility Implementation". Disponível em: http://www.dstc.edu.au/Products/CORBA/MOF/. June, 2001.

[4].Document Object Model (DOM) Level 1 Specification Version 1.0 (1998) -http://www.w3.org/TR/REC-DOM-Level-1/

[5].Document Object Model (DOM) Level 2 Core Specification version 1.0 (2000) - http://www.w3.org/TR/DOM-Level-2-Core/

[6].Document Object Model (DOM) Level 3 Core Specification Version 1.0 (2002) - http://www.w3.org/TR/DOM-Level-3-Core/

[7].JSR-40 Home Page. "Java Metadata Interface". Disponível em: http://java.sun.com/aboutJava/communityprocess/jsr/jsr_040_jolap.html. March, 2002.

[8].MARCO, David; INMON, W. H. "Building and Managing the Metadata Repository". 2000. New York. John Wiley & Sons, Inc.

[9].MEGGINSON, David et al. "SAX: The Simple API for XML". Disponível em: http://www.saxproject.org/.

[10].Object Management Group, "Meta Object Facility Specification, Version 1.3". Disponível em: http://www.dstc.edu.au/Research/Projects/MOF/rtf/. Veja também em: http://www.omg.org/. September, 1999.

[11].Object Management Group. "Unified Modeling Language Specification, Version 1.4". Disponível em: http://cgi.omg.org/docs/formal/01-09-67.pdf. September, 2001.

[12].Object Management Group. "XML Metadata Interchange Specification, Version 1.1". Disponível em: http://www.omg.org/. June, 2000.

[13].OMG Architecture Board MDA Drafting Team. "Model-Driven Architecture: A Technical Perspective". Disponível em: ftp://ftp.omg.org/pub/docs/ab/01-02-01.pdf. February, 2001.

[14].POOLE,John; CHANG,Dan; TOLBERT, Douglas; MELLOR, David. "Common Warehouse Metamodel: An Introduction to the Standard for Data Warehouse Integration". 2001. New York. John Willey & Sons, Inc.

[15].SANTOS, Hélio; BARROS, Roberto; FONSECA, Decio. "A Proposal for Management of RDF and RDF Schema Metadata in MOF". Proc. Int. Conf. on Ontologies, Databases and Applications of SEmantics (ODBASE'2003), Sicily, Italy, November 2003. Lecture Notes in Computer Science. Springer, 2003

[16].SANTOS, Hélio; BARROS, Roberto; FONSECA, Decio. "Uma Proposta para Gerenciamento de Metadados XML e DTD em MOF". Anais 18° Simpósio Brasileiro de Banco de Dados, Manaus, Brasil, Outubro, 2003.

[17].Sun Microsystems. "Metadata Repository Home". Disponível em: http://mdr.netbeans. org/. 2002.

[18].TANNENBAUM, Adrienne. "Metadata Solutions: Using Metamodels, Repositories, XML and Enterprise Portals to Generate Information on Demand". 2002. New York. Addison Wesley.

Comparación de un sistema de colonias de hormigas y una estrategia evolutiva para un Problema Multiobjetivo de Ruteo de Vehículos con Ventanas de Tiempo.

Augusto Hermosilla y Benjamín Barán
Universidad Nacional de Asunción, Centro Nacional de Computación,
San Lorenzo, Paraguay, Casilla de Correo 1439
ahermosilla@cnc.una.py, *bbaran@cnc.una.py*

Abstract

The present work compares an Ant Colony System (ACS) with an Evolutionary Strategy (variant of the Pareto Archived Evolutionary Strategy) utilized in solving a multiobjective vehicle routing problem with time windows (VRPTW). We analyze instances of different classes and sizes, widely studied in the literature. Computational results show that the ACS has a better performance than the Evolutionary Strategy, especially in instances of larger size.

Keywords: Multiobjective Optimization, Vehicle Routing Problem with Time Windows, Ant Colony System, Pareto Archived Evolutionary Algorithm.

Resumen

El presente trabajo compara un Sistema de Optimización basado en Colonias de Hormigas (*Ant Colony Optimization*) con una estrategia evolutiva (variante del *Pareto Archived Evolutionary Strategy*), utilizados en la resolución multiobjetivo del problema de ruteo de vehículos con ventanas de tiempo. (*Vehicle Routing Problem with Time Windows*, VRPTW). Se analizan problemas de diversos tipos y tamaños, ampliamente estudiados en la literatura. Resultados experimentales demuestran que el sistema de colonias de hormigas tiene un mejor desempeño en más tipos de problemas que la estrategia evolutiva, especialmente en problemas más grandes.

Palabras claves: Optimización multiobjetivo, Problema del Ruteo de Vehículos con Ventanas de Tiempo, Sistema de Colonias de Hormigas, Pareto Archived Evolutionary Algorithm.

1. Introducción

El problema del ruteo de vehículos (*Vehicle Routing Problem*, o VRP) es ya considerado un paradigma en la literatura especializada [4]. Este problema se plantea como un depósito central que cuenta con una flota de vehículos y debe atender a un conjunto de clientes geográficamente distribuidos. El objetivo del VRP es entregar bienes a este conjunto de clientes con demandas conocidas, al mínimo coste, encontrando las rutas óptimas que se originan y terminan en el referido depósito. Cada cliente es servido una sola vez y todos los clientes deben ser atendidos, para lo cual se los asigna a vehículos que llevarán la carga (demanda de los clientes que visitará) sin exceder su capacidad máxima de transporte.

Para extender el VRP tradicional agregando la restricción adicional de asociar una ventana de tiempo a cada cliente, se define un intervalo en el que cada cliente debe ser atendido y se obtiene el problema del ruteo de vehículos con ventanas de tiempo (*Vehicle Routing Problem with Time Windows*, o VRPTW) [22]. Al considerar estas ventanas de tiempo, el costo total de ruteo y planeamiento (*scheduling*) incluyen: la distancia total recorrida que está asociada al tiempo total de viaje efectivo, el tiempo total de espera en que se incurre cuando un vehículo llega muy temprano a la ubicación de un cliente y el tiempo total de servicio (tiempo para descargar todas las mercaderías solicitadas por cada cliente). Claramente, el tiempo en el que se inicia el servicio a un cliente debe ser mayor o igual al inicio de su ventana de tiempo y el instante en que se llega a cada cliente debe ser menor o igual al fin de su ventana de tiempo. Si un vehículo llega a la ubicación de un cliente antes del inicio de su ventana de tiempo, debe esperar hasta esa hora para servir a ese cliente.

El problema espacial del ruteo de vehículos ha sido extensamente estudiado en la literatura. Una gran variedad de abordajes ha sido ya publicado para resolver este problema, utilizando implementaciones paralelas [15], estrategias híbridas combinando métodos de búsqueda local con algoritmos evolutivos [21], redes neuronales [12, 26], una heurística que utiliza información de feromonas [20] y trabajos que resuelven este problema con estrategias evolutivas para optimizar la demanda de cada vehículo además de la optimización del número de vehículos y el tiempo total de viaje [23].

La restricción de ventanas de tiempo solo ha sido considerada recientemente y varios abordajes han sido presentados para resolver el VRPTW: algoritmos paralelos con un número polinomial de procesadores [13], algoritmos genéticos [8, 16, 17, 24], recocido simulado paralelo [1, 10] y colonias múltiples de hormigas [11]. Estos trabajos analizan el problema en cuestión en un contexto mono-objetivo, aunque en algunos casos se proponen priorizaciones y/o combinaciones de objetivos, sin llegar a calcular el conjunto completo de soluciones Pareto. Estudios de las principales heurísticas y metaheurísticas pueden ser encontrados, por ejemplo, en [6] y [7].

Por su parte, el presente trabajo resuelve el problema del VRPTW en un contexto multiobjetivo, considerando dos técnicas evolutivas que han demostrado su capacidad de resolver adecuadamente problemas de similar complejidad:

(1) las optimizaciones con colonias de hormigas (ACO), en la versión conocida como MOACS-VRPTW, propuesta por Barán y Schaerer [3], a partir de una optimización del MACS-VRPTW publicado por Gambardella et al. [11]; y

(2) los algoritmos evolutivos multiobjetivos (*Multi-Objective Evolutionary Algorithms* – MOEAs) en una variante propuesta en este trabajo que daremos en llamar PAES-DRI, obtenida a partir de la propuesta ES-DRI presentada por Mester [19].

Esta comparación experimental puede ser considerada como una continuación del trabajo presentado en [2].

El resto del trabajo es organizado de la siguiente manera: la sección 2 presenta la formulación tradicional del problema, la sección 3 contiene la formulación multiobjetivo, la sección 4 resume el MOACS-VRPTW, la 5 explica el PAES-DRI, la 6 presenta las métricas de desempeño utilizadas para la comparación, mientras la sección 7 presenta resultados experimentales. Finalmente, la sección 8 presenta las conclusiones del trabajo.

2. Formulación Matemática del VRPTW

El VRPTW es un problema combinatorial que puede ser formulado matemáticamente como un grafo dirigido ***G(V,A)***. El problema considera un conjunto de vértices $V = \{v_0, v_1, \ldots, v_n\}$, donde v_0 representa el depósito, mientras que v_i *(i = 1, ..., n)* representa a cada uno de los *n* clientes (o ciudades) a ser visitados. Por otro lado, el conjunto de arcos está dado por: $A = \{(v_i, v_j) / v_i, v_j \in V; i \neq j\}$ [19].

Por su parte, $C = \{c_{ij}\} \in \Re^{(n+1)x(n+1)}$ es una matriz de distancias o costos no negativos entre cada par de vértices v_i y v_j (incluyendo depósito y clientes).

$q \in \Re^n$ es el vector de las demandas de los clientes y q_i representa la cantidad de bienes requeridos por el cliente v_i.

m representa el número de vehículos, que se asumen todos idénticos, y con una capacidad *Q* por vehículo. A cada vehículo *i* se le asignará una ruta R_i.

$[e_i, l_i]$ representa la ventana de tiempo del cliente v_i; con e_i como la hora más temprana y l_i como la hora más tardía en que se puede iniciar el servicio a dicho cliente v_i. Por su parte, δ_i representa el tiempo requerido en descargar la cantidad de mercaderías q_i en el cliente v_i.

Para este trabajo, se considera sólo el problema simétrico con $c_{ij} = c_{ji}$ para cada $(v_i, v_j) \in A$. En este caso, es común reemplazar *A* por el conjunto $E = \{(v_i, v_j) \mid v_i, v_j \in V; i < j\}$. Sin pérdida de generalidad, y por simplicidad, se asume que el tiempo t_{ij} necesario para ir de v_i a v_j es igual a la distancia c_{ij} (se asume velocidad constante e unitaria para los vehículos). El intervalo $[e_0, l_0]$ en el depósito es conocido como horizonte de planeación (*scheduling horizon*).

Definiendo b_v como el instante de inicio del servicio al cliente v, se puede escribir la condición de factibilidad que debe cumplir una ruta $R_i = \{v_0^i, v_1^i, \ldots, v_{k_i}^i, v_{k_i+1}^i\}$, donde $v_j^i \in V$, $v_0^i = v_{k_i+1}^i = 0$ (el 0 denota el depósito) y k_i es la cantidad de clientes atendidos en la ruta i:

por la capacidad del vehículo: $\sum_{j=1}^{k_i} q_j^i \le Q. \quad i = 1, \ldots, m$ **(1)**

por las ventanas de tiempo: $e_{v_j^i} \le b_{v_j^i} \le l_{v_j^i}. \quad i = 1, \ldots, m; \quad j = 1, \ldots, k_i + 1$ **(2)**

Asumiendo que todo vehículo i viaja al próximo cliente v_j^i una vez que haya terminado de servir al cliente actual v_{j-1}^i, el valor de $b_{v_j^i}$ puede ser calculado recursivamente de la siguiente manera:

$$b_{v_j^i} = \max\{e_{v_j^i}; b_{v_{j-1}^i} + \delta_{v_{j-1}^i} + c_{v_{j-1}^i, v_j^i}\}, \qquad \text{con } b_0^i = e_0 \text{ y } \delta_0^i = 0. \quad \textbf{(3)}$$

Para este trabajo, se considera $e_0 = 0$, es decir, todos los vehículos parten del depósito en el mismo instante.

Así, el tiempo de espera en el cliente v_j^i puede ser definido como:

$$w_{v_j^i} = \max\{0; b_{v_j^i} - b_{v_{j-1}^i} - \delta_{v_{j-1}^i} - c_{v_{j-1}^i, v_j^i}\}. \quad \textbf{(4)}$$

El costo de la ruta R_i puede estar dado por la duración total de la misma, en cuyo caso estaría definido como:

$$T(R_i) = \sum_{j=0}^{k_i} c_{v_j^i, v_{j+1}^i} + \sum_{j=1}^{k_i} \delta_{v_j^i} + \sum_{j=0}^{k_i} w_{v_j^i}. \quad \textbf{(5)}$$

Tradicionalmente, el objetivo primario es la minimización del tamaño *m* de la flota de vehículos y el objetivo secundario es la minimización de la distancia total recorrida (tiempo total de viaje) o la duración total de las rutas (tiempo total de entrega), [6, 7].

3. Formulación del VRPTW como un Problema Multiobjetivo

Como ya fuera discutido, el VRPTW es tradicionalmente formulado como un problema bi-objetivo lexicográfico [25]. En contrapartida, este trabajo utiliza una formulación multiobjetivo presentada en [3]. En este contexto, la función objetivo es un vector tridimensional $F = [F_1, F_2, F_3]^T$, y ningún objetivo es más importante que los otros. Los objetivos propuestos son:

- $F_1 = m$ ... es el número de vehículos (o tamaño de la flota); el cual es una medida de la inversión inicial necesaria y el costo del mantenimiento de vehículos.
- $F_2 = \sum_{i=1}^{m} \sum_{j=0}^{k_i} c_{v_j^i, v_{j+1}^i}$ es el tiempo total de viaje (o distancia total recorrida), que representa el consumo de combustible.
- $F_3 = F_2 + \sum_{i=1}^{m} \sum_{j=1}^{k_i} \delta_{v_j^i} + \sum_{i=1}^{m} \sum_{j=0}^{k_i} w_{v_j^i}$ es el tiempo total de entrega y se relaciona con el gasto necesario en los viáticos de los conductores y el seguro de los vehículos.

donde v_j^i denota el cliente servido en el *j*-ésimo orden de la ruta R_i y k_i representa el número total de clientes servidos en dicha ruta R_i.

Otros objetivos importantes y que pueden ser implementados en trabajos futuros son:

- $F_4 = max_i \left(b_{v_{k_i+1}}^i \right)$ es la duración de la ruta más larga e indica la satisfacción de los clientes y la posibilidad de asignar los vehículos a otras tareas.
- $F_5 = \sum_{i=1}^{m} \sum_{j=0}^{k_i} w_{v_j^i}$ es el tiempo total de espera, que puede ser reducido admitiendo $b_0^i \geq e_0$ en (4). Este objetivo se relaciona con el riesgo y los viáticos adicionales en los que se incurren debido a llegar a la ubicación de un cliente antes del inicio de su ventana de tiempo.

En general un problema de optimización multiobjetivo puede formularse como:

Minimizar $[f_1(x), f_2(x), \ldots, f_K(x)]^T$

Sujeto a $x \in S$

donde K es el número de objetivos, $x = (x_1, x_2, \ldots, x_l)^T$ es un vector con l variables de decisión, y S es la región factible (o conjunto de soluciones factibles).

Para comparar dos soluciones $x, y \in S$, es frecuente utilizar el concepto de dominancia [25]. En un contexto de minimización, se dice que una solución x domina a la solución y, denotado por $x \succ y$, si:

a. x es por lo menos igual a y en cada objetivo: $F_k(x) \leq F_k(y), \forall k \in \{1, \ldots, K\}$

b. x es mejor que y en por lo menos un objetivo: $\exists k \in \{1, \ldots, K\}, \quad F_k(x) < F_k(y)$.

Dado que ningún objetivo se considera más importante que los demás, se trata de un contexto realmente multiobjetivo donde en principio, la solución puede no ser única, sino un conjunto de soluciones óptimas, no comparables entre sí, conocido como *Conjunto Pareto P*. La imagen de *P* en el contra-dominio de las funciones objetivo es conocida como Frente Pareto *FP* [25].

4. Algoritmo MOACS-VRPTW

Barán y Schaerer [3] propusieron una variación del Algoritmo de Colonias Múltiples de Hormigas para el VRPTW (*Multiple Ant Colony System for Vehicle Routing Problems with Time Windows, MACS-VRPTW*), originalmente propuesto por Gambardella et al. [11]. Esta variación fue denominada Colonia de Hormigas para

Optimización Multiobjetivo del VRPTW (*Multiobjective Ant Colony System for Vehicle Routing Problem with Time Windows*, *MOACS-VRPTW*). Esta metaheurística utiliza una sola colonia de hormigas para minimizar simultáneamente las tres funciones objetivo, presentadas en la sección anterior. Todas las funciones comparten los mismos rastros de feromonas. De esta manera, el conocimiento de buenas soluciones es igualmente importante para cada función objetivo. El referido MOACS-VRPTW puede ser resumido con el siguiente pseudocódigo:

Pseudocode MOACS-VRPTW

/* *Inicialización* */

Repeat /* *Ciclo principal* */

for each ant $k \in \{1,\ldots, m\}$ **do**
Construir una solución (ψ^k)
if $\psi^k \in P$ /* si solución encontrada forma parte del conjunto Pareto */
grabar ψ^k y borrar soluciones dominadas de *P*
end if
end for
/* Actualizar feromonas τ */
if $\tau_{0'} > \tau_0$ /* Un mejor *FP* ha sido encontrado */
Reinicializar rastros $\{\tau\}$ con τ_0
else
for each $\psi^P \in P$ do
Actualización global de feromonas
end for
end if

until criterio de parada ha sido satisfecho.

5. Algoritmo PAES-DRI

El presente trabajo, propone una nueva variante de MOEA que damos en llamar PAES-DRI, inspirada en el ES-DRI originalmente propuesto por Mester [18]. El ES-DRI (*Evolutionary Strategy - Dichotomy Remove-Insert*) es un algoritmo heurístico muy efectivo para determinar la solución de versiones en gran escala del VRPTW. Este algoritmo está basado en las ideas de estrategias evolutivas (1+1) y algunos operadores de mutación que trabajan con un esquema dicotómico de remover-insertar. Por su parte, el PAES (*Pareto Archived Evolutionary Strategy*) [14] es un esquema simple de evolución para problemas multiobjetivo. Este algoritmo es una estrategia evolutiva (1+1) que utiliza búsqueda local en una población de un individuo, pero usando un archivo de referencia de soluciones previamente encontradas a fin de identificar el *ranking* de dominancia aproximado de las soluciones *actual* y *candidata*. El PAES-DRI, propuesto en este trabajo, puede ser resumido con el siguiente pseudocódigo:

Pseudocode PAES-DRI

Generar una solución inicial *a* y agregarla al conjunto Pareto P

Repeat /* *Ciclo principal* */
Mutar a para obtener b y evaluar b
if ($a \succ b$) /* si la solución a domina a b */
Descartar b
else if ($b \succ a$)
$a \leftarrow b$
Agregar a al conjunto Pareto *P* y eliminar las soluciones dominadas de *P*
else if (algún elemento de *P* domina a b)
Descartar b
else (Determinar cual será la solución actual a y si se debe agregar b a *P*)
end if

until un criterio de parada ha sido alcanzado.

Considerando que esta variante está siendo propuesta en este trabajo, se la describe en mayores detalles en las siguientes subsecciones, que están organizadas de la siguiente manera: la subsección 5.1 resume la heurística de inicialización y la 5.2 presenta el operador de mutación empleado en este MOEA.

5.1. Inicialización

La inicialización de una solución es realizada utilizando una heurística basada en la inserción más económica [5]. Las rutas son construidas una a la vez en forma secuencial. Para esto, es necesario determinar los nodos que inicialicen las rutas. Estos nodos son denominados nodos *semillas* (*seed nodes*). Se selecciona un nodo como inicio de la primera ruta y se procede iterativamente a insertar nodos en esa ruta. Cuando no se pueden realizar más inserciones factibles, se selecciona otro nodo semilla para inicializar otra ruta y se procede iterativamente hasta atender a todos los clientes

5.2. Operador de mutación

Tal como se había mencionado con anterioridad, el PAES-DRI es una variación del ES-DRI [19]. El operador de mutación utilizado en dicho trabajo genera una nueva solución removiendo e insertando β clientes de la solución actual a para obtener una solución candidata b. Este procedimiento genera nuevas rutas factibles en un proceso de dos fases que construye y mejora nuevas soluciones. El número de clientes a ser removidos de la solución actual es determinado en [19] conforme a la ecuación:

$$\beta = (0.1 + 0.5rnd)\,n\,. \qquad (6)$$

donde *n* es siempre el número total de clientes y *rnd* es un número generado aleatoriamente y distribuido uniformemente entre 0 y 1.

Después de remover un cliente de la solución actual se procede a insertarlo en otra posición para obtener una solución candidata factible. La inserción se realiza en forma secuencial. Para insertar el cliente v_u se determinan todos los lugares factibles de inserción y se halla para cada uno de ellos el costo de inserción de acuerdo a la función vectorial $[\Delta F_1, \Delta F_2, \Delta F_3]$. Como la función de inserción tiene tres dimensiones, se determinan todas las opciones de inserción que no son dominadas y se selecciona una al azar.

6. Métricas de desempeño

Para evaluar los resultados experimentales de los dos algoritmos arriba presentados, se seleccionaron cuatro métricas, considerando que no existe una única métrica que pueda por sí sola medir el desempeño, eficiencia y efectividad de los algoritmos evolutivos multiobjetivos [25]. Las métricas utilizadas para el presente trabajo son las siguientes:

- *Overall Non-dominated Vector Generation* (ONVG): que simplemente cuenta el número de soluciones en el Frente Pareto calculado, denotado como Y_{known}

$$ONVG \overset{\Delta}{=} |\,Y_{known}\,|_c\,. \qquad (8)$$

  donde $|\ |_c$ denota cardinalidad. A mayor valor del ONVG, se tiene un mejor conocimiento de los detalles del Frente Pareto.

- *Generational Distance* (*GD*) representa que tan alejado se encuentra $\boldsymbol{Y}_{known}$ del Frente Pareto Óptimo denotado como $\boldsymbol{Y}_{true}$. Como $\boldsymbol{Y}_{true}$ no es conocido en teoría, debe estimarse haciendo muchas corridas de diversos algoritmos multiobjetivos para el mismo problema, escogiendo las soluciones Pareto óptimas encontradas con la unión de los resultados obtenidos con todos los experimentos. Claramente, un menor valor de GD, indica que un conjunto $\boldsymbol{Y}_{known}$ es mejor que otro. La distancia generacional se define como:

$$GD \overset{\Delta}{=} \frac{\left(\sum_{i=1}^{N} d_i^2\right)^{1/2}}{N}\,. \qquad (9)$$

  donde N es el número de vectores en $\boldsymbol{Y}_{known}$, y d_i es la distancia euclidiana (en el espacio objetivo) entre cada vector y el miembro *más cercano* en $\boldsymbol{Y}_{true}$.

- *Coverage* (*C*): cuenta el número de soluciones del Frente Pareto de un algoritmo 1, denotado como $\boldsymbol{Y}_{known1}$, que son dominadas por alguna solución del Frente Pareto del Algoritmo 2 ($\boldsymbol{Y}_{known2}$). Lógicamente, al comparar dos algoritmos, el que tenga un menor valor de *C*, será superior.

- *Spacing (S)*: mide el rango de la variancia de los vectores vecinos en $\boldsymbol{Y}_{known}$. Esta métrica está definida como:

$$S \stackrel{\Delta}{=} \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(\bar{d} - d_i)^2} \qquad (\mathbf{10})$$

donde $d_i = min_j\left(\left|f_1^i(x) - f_1^j(x)\right| + \left|f_2^i(x) - f_2^j(x)\right| + \left|f_3^i(x) - f_3^j(x)\right|\right);\ i, j = 1,2,\ldots,N;\ \bar{d}$ es la media de todos los d_i y N es el número de vectores en $\boldsymbol{Y}_{known}$. Si el *spacing* es igual a cero, entonces los vectores están uniformemente distribuidos. De esta manera, es preferible un menor valor de *spacing*.

7. Resultados Experimentales

Los problemas de prueba forman parte de un conjunto de problemas ya clásicos y ampliamente estudiados en la literatura [22]. Se han considerado diversos tamaños del problema, a fin de poder comparar el desempeño de los algoritmos.

Se han realizado diez corridas de cada algoritmo, formando los Frentes Pareto $\boldsymbol{Y}_{MOACS}$ y $\boldsymbol{Y}_{PAES\text{-}DRI}$. Como no se conoce la solución teórica óptima del problema, se aproximó $\boldsymbol{Y}_{true}$. mediante la unión de todas las soluciones obtenidas por los investigadores, utilizando los 2 algoritmos que se están comparando, y eliminando de este conjunto las soluciones dominadas.

El número de generaciones del PAES-DRI es de 1000 y el de MOACS-VRPTW es de 3000. De esta manera pueden encontrarse tiempos similares de ejecución.

En la Tabla 1 se pueden observar los resultados experimentales para instancias de 25 clientes y se pintan en gris los mejores valores experimentales para cada métrica. Con respecto a ONVG, se puede apreciar que ningún algoritmo domina realmente al otro, dado que PAES-DRI es superior a MOACS-VRPTW en las instancias C207_25, R101_25 y RC105_25. Por su parte, MOACS-VRPTW es superior a PAES-DRI en las instancias C101_25, R210_25, RC202_25.

En cuanto a las métricas GD y C, podemos observar que PAES-DRI obtiene mejores soluciones que MOACS-VRPTW en 5 de 6 instancias. De esta forma podemos afirmar que las soluciones generadas por PAES-DRI utilizan menor cantidad de vehículos y recorren una menor distancia en menor tiempo.

Por otro lado, la distribución de las soluciones obtenidas por MOACS-VRPTW es más uniforme que las generadas por PAES-DRI (ver métrica S). Pero este resultado sólo indica que MOACS-VRPTW obtiene soluciones más diversas que PAES-DRI.

Ahora podemos apreciar que PAES-DRI tiene un desempeño superior a MOACS-VRPTW en las instancias C101_25, C207_25, R101_25 y RC105_25, considerando simultáneamente las cuatro métricas.

En la Tabla 2 se pueden observar los resultados experimentales para instancias de 50 clientes. MOACS-VRPTW obtiene una mayor cantidad de soluciones no dominadas que PAES-DRI (ver métrica ONVG) en 4 de 6 instancias. Con respecto a las métricas GD y C, MOACS-VRPTW obtiene mejores soluciones que PAES-DRI en 4 de 6 instancias. Por otro lado, ningún algoritmo domina al otro en la generación de soluciones uniformemente distribuidas, ya que MOACS-VRPTW es superior en las instancias C101_50, R210_50 y RC105_50. Así mismo, PAES-DRI es superior en las instancias C207_50, R101_50 y RC202_50.

Si consideremos simultáneamente las cuatro métricas, se puede observar que MOACS-VRPTW tiene un desempeño superior a PAES-DRI en las instancias C101_50, C207_50, R210_50 y RC202_50.

Tabla 1. Métricas de desempeño para los dos algoritmos comparados en el presente trabajo en instancias de 25 clientes.

Instancia	MOEA	ONVG	GD	C	S
C101_25	MOACS	8	0.1045	2	0.1539
	PAES-DRI	5	0.0000	8	0.1610
C207_25	MOACS	11	0.3830	0	0.1180
	PAES-DRI	20	0.0000	11	0.0486
R101_25	MOACS	6	0.1329	2	0.1077
	PAES-DRI	8	0.0000	6	0.1617
R210_25	MOACS	14	0.0866	0	0.1375
	PAES-DRI	6	0.0000	6	0.3810
RC105_25	MOACS	2	0.9664	0	0.0000
	PAES-DRI	10	0.0000	2	0.1009
RC202_25	MOACS	22	0.0353	15	0.0417
	PAES-DRI	17	0.3584	3	0.0819

Tabla 2. Métricas de desempeño para los dos algoritmos comparados en el presente trabajo en instancias de 50 clientes.

Instancia	MOEA	ONVG	GD	C	S
C101_50	MOACS	5	0.0000	1	0.2805
	PAES-DRI	1	0.5840	0	0.0000
C207_50	MOACS	18	0.0000	18	0.1281
	PAES-DRI	18	0.2961	0	0.0739
R101_50	MOACS	8	0.5701	0	0.1466
	PAES-DRI	7	0.0000	8	0.1345
R210_50	MOACS	16	0.0000	4	0.0606
	PAES-DRI	4	0.1683	0	0.1869
RC105_50	MOACS	2	0.0000	0	0.0000
	PAES-DRI	3	0.0000	0	0.4750
RC202_50	MOACS	22	0.0000	13	0.1250
	PAES-DRI	19	0.2661	0	0.0858

En la Tabla 3 se pueden observar los resultados experimentales para instancias de 100 clientes. PAES-DRI obtiene una mayor cantidad de soluciones no dominadas que MOACS-VRPTW (ver métrica ONVG) en 5 de 6 instancias. En cuanto a las métricas GD y C, MOACS-VRPTW obtiene en general mejores soluciones que PAES-DRI en 4 de 6 instancias. Así mismo, podemos observar que MOACS-VRPTW es ligeramente superior a PAES-DRI en la generación de soluciones uniformemente distribuidas (ver métrica S).

De igual manera a lo procedido anteriormente, podemos considerar simultáneamente las cuatro métricas y afirmar que MOACS-VRPTW tiene un desempeño superior a PAES-DRI en las instancias C101_100, C207_100, R101_100, R210_100 y RC202_100.

Si observamos el desempeño global de los algoritmos, podemos resaltar que el MOACS-VRPTW obtiene mejores resultados en las instancias de tipo C1, C2, R2, RC2 y el PAES-DRI en R1 y RC1. Esto sugiere la necesidad de implementar una heurística en el PAES-DRI que optimice localmente cada ruta, a fin de mejorar su convergencia. Este hecho es resaltado en [9], dado que los problemas de clase C2, R2 y RC2 se caracterizan por tener soluciones en que el número de vehículos es pequeño (2-4 vehículos) y la cantidad de clientes servidos en cada ruta es relativamente grande (alrededor de 30 clientes atendidos por ruta).

Tabla 3. Métricas de desempeño para los dos algoritmos comparados en el presente trabajo en instancias de 100 clientes.

Instancia	MOEA	ONVG	GD	C	S
C101_100	MOACS	1	0.0000	2	0.0000
	PAES-DRI	2	1.2961	0	0.0000
C207_100	MOACS	1	0.0000	15	0.0000
	PAES-DRI	15	1.4841	0	0.0653
R101_100	MOACS	14	0.4224	0	0.0926
	PAES-DRI	7	0.0000	14	0.2010
R210_100	MOACS	7	0.0000	19	0.3528
	PAES-DRI	19	0.6004	0	0.0761
RC105_100	MOACS	3	0.3767	0	1.3818
	PAES-DRI	4	0.0000	1	0.2916
RC202_100	MOACS	10	0.0000	10	0.1657
	PAES-DRI	16	0.1291	0	0.4165

8. Conclusión

El problema del ruteo de vehículos con ventanas de tiempo va ganando importancia en la medida que el mundo globalizado fuerza a las empresas distribuidoras modernas a ser cada vez más eficientes e incrementar su región de trabajo, para mantenerse competitivas. Como natural consecuencia, crece el interés en el área y se van proponiendo nuevos abordajes, entre los que se destacan los algoritmos evolutivos, por su flexibilidad para analizar circunstancias cambiantes y objetivos contradictorios. A raíz de esto, resulta interesante comparar las nuevas alternativas que se van proponiendo para lograr aplicar el abordaje más adecuado a cada realidad.

Es en este contexto que el presente trabajo propone una variante de algoritmo evolutivo multiobjetivo y realiza una comparación entre un sistema de colonias de hormigas y una estrategia evolutiva para el problema del ruteo multiobjetivo de vehículos con ventanas de tiempo.

Los experimentos demuestran que el MOACS-VRPTW tiene un mejor desempeño en más tipos de problemas, pero es necesaria la implementación de una heurística en el PAES-DRI que optimice localmente cada ruta, a fin de mejorar la calidad de las soluciones a los problemas de clase C2, R2 y RC2.

Los autores proponen diversos trabajos futuros, tales como: una comparación con los mejores algoritmos mono-objetivos publicados; implementar otras variantes de algoritmos multiobjetivos utilizando manejo de restricciones y resolver otras variantes del problema de ruteo de vehículos considerando objetivos adicionales, como los propuestos en la sección 3.

Referencias

[1] Arbelaitz, O., Rodriguez, C. y Zamkola, I. Low cost parallel solutions for the VRPTW optimization problem. *Fourth International Conference on Parallel Processing Workshops*, 2001, pp.176-181.

[2] Barán B. y Hermosilla, A. Comparación de un Sistema de Colonias de Hormigas y una Estrategia Evolutiva para el Problema del Ruteo de Vehículos con Ventanas de Tiempo en un Contexto Multiobjetivo. *IX Jornadas Iberoamericanas de Informática*, Cartagena de Indias, Colombia. 2003.

[3] Barán, B. y Schaerer, M. A multiobjective Ant Colony System for Vehicle Routing Problem with Time Windows. *Proceedings of the 21st IASTED International Conference APPLIED INFORMATICS.* Innsbruck, Austria. 2003.

[4] Bodin, L. Golden, B., Assad, A. y Ball, M. Routing and Scheduling of Vehicles and Crews. The State of the Art. *Computers and Operations Research.*Vol. 10, 1983, pp.62-212.

[5] Bräysy, O. A reactive variable neighborhood for the Vehicle Routing Problem with Time Windows. 2001. To appear in *Informs Journal on Computing*.

[6] Bräysy, O. y Gendreau, M. Route construction and local search algorithms for the vehicle routing problem with

time windows, Report STF42 A01024, App. Mathematics, Dep. of Optimization, Norway. 2001.

[7] Bräysy, O. y Gendreau, M. Metaheuristics for the vehicle routing problem with time windows, Internal Report STF42 A01025, SINTEF Applied Mathematics, Department of Optimization, Norway. 2001.

[8] Chin, A., Kit, H. y Lim, A. A new GA approach for the vehicle routing problem. *11th IEEE International Conference on Tools with Artificial Intelligence*, 1999, pp.307-310.

[9] Cordeau, J.-F., Laporte, G. y Mercier, A. A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows. *Journal of the Operational Research Society*. Vol. 52, 2001, pp.928-936.

[10] Czech, Z.J. y Czarnas, P. Parallel simulated annealing for the vehicle routing problem with time windows. *10th Euromicro Workshop on Parallel, Distributed and Network-based Proceedings*, 2002, pp.376-383.

[11] Gambardella, L., Taillard, E. y Agazzi, G. *News ideas in optimization*. Mac Graw-Hill: London, 1999, pp.73-76.

[12] Gomes,L. y von Zuben, F.J. A neuro-fuzzy approach to the capacitated vehicle routing problem. *International Joint Conference on Neural Networks*. Vol. 2, 2002, pp.1930-1935.

[13] Gupta, A. y Krishnamurti, R. Parallel algorithms for vehicle routing problems. *Fourth International Conference on High-Performance Computing*, 1997, pp.144-151.

[14] Knowles, J. and Corne, D. The Pareto Archived Evolution Strategy: A New Baseline Algorithm for Pareto Multiobjective Optimisation. Dept. of Computer Science. University of Reading. UK. 1999.

[15] Lau, K.K., Kumar, M.J. y Achuthan, N.R. Parallel implementation of branch and bound algorithm for solving vehicle routing problem on NOWs. *3rd Internationl Symposium on Parallel Architectures, Algorithms, and Networks*, 1997, pp.247-253.

[16] Louis, S.J., Xiangying, Y. y Zhen, Y.Y. Multiple vehicle routing with time windows using genetic algorithms. *Congress on Evolutionary Computation*, 1808(3), 1999.

[17] Maeda, O., Nakamura, M., Ombuki, B.M. y Onaga, K. A genetic algorithm approach to vehicle routing problem with time deadlines in geographical information systems. *International Conference on Systems, Man, and Cybernetics*. Vol. 4, 1999, pp.595-600.

[18] Mester, D. The Parallel algorithm for Vehicle Routing Problem with Time Windows restrictions. Scientific Report, Minerva Optimization Center, Technion, Israel. 1999.

[19] Mester, D. An evolutionary strategies algorithm for large scale vehicle routing problem with capacitate and time windows restrictions, Institute of Evolution, Mathematical and Population Genetics, Univ. of Haifa, Israel. 2002.

[20] Murao, H., Tohmata, K., Konishi, M. y Kitamura, S. Pheromone based transportation scheduling system for the multi-vehicle routing problem *IEEE Int. Conference on Systems, Man, and Cybernetics*. Vol. 4, 1999, pp.430-434.

[21] Pedroso, J.P. Niche search: An application in vehicle routing. *IEEE World Congress on Computational Intelligence*, 1998, pp.177–182.

[22] Solomon, M.M. Algorithms for Vehicle Routing and Scheduling Problems with time window constraints. Northeastern University, Boston, Massachusetts, (December, 1985).

[23] Takeno, T., Tsujimura, Y. y Yamazaki, G. A single-phase method based on evolution calculation for vehicle routing problem. *4th Int. Conf. on Computational Intelligence and Multimedia Applications*, 2001, pp.103–07.

[24] Tan, K.C., Lee, T.H., Ou, K. y Lee, L.H. A messy genetic algorithm for the vehicle routing problem with time window constraints. *Congress on Evolutionary Computation*, Vol. 1, 2001, pp.679-686.

[25] Van Veldhuisen, D. A. Multiobjective Evolutionary Algorithms: Classifications, Analyses and New Innovations. PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology. Ohio, USA. (May, 1999).

[26] Yoshiike, N.y Takefuji, Y. Vehicle routing problem using clustering algorithm by maximum neural networks. *2nd Int. Conf. on Intelligent Processing and Manufacturing of Materials*. Vol. 2, 1999, pp.1109-1113.

Modelos Automatizables de Estimación muy Temprana del Tiempo y Esfuerzo de Desarrollo de Software de Gestión

Pedro F. Salvetto
Universidad ORT Uruguay, Laboratorio de Investigación de Sistemas de Información,
Montevideo, Uruguay, Cuareim 1451, 11100
salvetto@ort.edu.uy
Juan C. Nogueira
Universidad ORT Uruguay, Laboratorio de Investigación de Sistemas de Información,
Montevideo, Uruguay, Cuareim 1451, 11100
nogueira@ort.edu.uy
Javier Segovia
Universidad Politécnica de Madrid
Departamento de Lenguajes Sistemas Informáticos e Ingeniería de Software
Campus de Montegancedo
28660 Boadilla del Monte (Madrid)
Facultad de Informática
fsegovia@fi.upm.es

Abstract

Contrary to most industrial production processes, software production processes generate intangible products and require intensive communication and coordination, which contributes to increase the risks and to complicate estimation. In spite of many years of research and development, formal and structured estimation (independent of expert judgment) of the time and effort required to develop a Management Information System (MIS) project remains an open problem The most extended estimation techniques at present time are supported by the premise - not so realistic - of stability of
requirements, and require human experts. Current models of estimation are based on metrics available in the early design phase. In this work we define an early MIS complexity metric and introduce formal models, ready to be automated, for estimating time and effort of MIS development projects. These models use as input parameters the development team efficiency, the requirements volatility, the speed of development and the complexity of the system to be developed. The complexity is measured automatically from the user's data views of the system,
with independence of the technology used. These models are applicable continuously during the project, from the very beginning at the requirements engineering phase up to later stages, and do not deny but assume the inevitable changes in requirements, supporting their management.

Keywords: software complexity metrics, early time and effort estimation, data and knowledge oriented development, software measure automation, empirical models

Resumen

A diferencia de los procesos de producción industrial, los procesos de producción de software generan productos intangibles y requieren comunicación y coordinación intensivas lo que contribuye a aumentar los riesgos y dificultar la estimación.
A pesar de largos años de investigación y desarrollo el problema de la estimación formal y estructurada (independiente del juicio experto) del tiempo y esfuerzo requeridos para desarrollar un Sistema de Gestión (SG) permanece abierto. Las técnicas de estimación más extendidas actualmente se apoyan en la premisa - poco realista - de estabilidad de requerimientos, y requieren expertos humanos. Los modelos de estimación actuales, se basan en métricas disponibles recién en la fase de diseño temprano del sistema.
En este trabajo se define una métrica temprana de complejidad de un SG y se presentan modelos formales (automatizables) de estimación del tiempo y esfuerzo de desarrollo de sistemas de información. Estos modelos emplean como parámetros de entrada la eficiencia del grupo de desarrollo, la volatilidad de los requerimientos, la velocidad de desarrollo y la complejidad del sistema a desarrollar. La complejidad es medida **automáticamente a partir de las vistas de datos de usuario del sistema con independencia de la tecnología** a **utilizar.** Estos modelos son aplicables continua y muy tempranamente desde la etapa de ingeniería de requerimientos y no desconocen los **inevitables** cambios en los requerimientos, sino que los asumen y apoyan su gestión.
Palabras clave: métricas de complejidad del software, estimación temprana de tiempo y esfuerzo, desarrollo orientado a datos y conocimiento, automatización de la medición del software, modelos empíricos

1 INTRODUCCIÓN

1.1 Descripción del problema

A diferencia de los procesos de producción industrial, los procesos de producción de software generan productos intangibles y requieren comunicación y coordinación intensivas lo que contribuye a aumentar los riesgos y dificultar la estimación.

A pesar de la existencia de modelos de estimación como COCOMO [5], COCOMO II [4] , su variante en desarrollo COSYSMO [25] y SLIM [21] entre otros, es muy común que los presupuestos y los tiempos de desarrollo sean subestimados [3][4][7][20].Estos modelos se caracterizan por incluir numerosos parámetros que deben ser estimados mediante juicio experto y parecen funcionar mucho mejor en manos de sus creadores, de expertos altamente entrenados en su uso y/o de quienes disponen de bases de datos con información histórica para calibrarlos. Por requerir juicio experto es posible que, observando un mismo escenario, distintos evaluadores obtengan estimaciones significativamente diferentes. En contraste con los avances en herramientas CASE, la gerencia de proyectos de software no ha avanzado sustancialmente. Para contribuir a la disciplina es necesario investigar el problema de la gerencia con un enfoque formal [19].

Algunas de las características del desarrollo de software que contribuyen a explicar la dificultad para construir modelos de estimación son a) el desarrollo de software requiere trabajo intelectual e interacción humana; b) a pesar del nivel repetible del CMM propuesto por el SEI, los procesos de desarrollo de software repetibles son raros; c) los proyectos de desarrollo de software pueden ser replicados, pero no repetidos; d) la tecnología cambia con tanta rapidez que es difícil, e incluso propenso a errores, incorporar la experiencia pasada ya que podemos estar evaluando una situación a partir de premisas no aplicables a ella.

A pesar de largos años de investigación y desarrollo el problema de la estimación formal y estructurada (independiente del juicio experto) del tiempo y esfuerzo requeridos para desarrollar un sistema de gestión permanece abierto. Las técnicas de estimación más extendidas actualmente se apoyan en la premisa - poco realista - de estabilidad de requerimientos, y requieren expertos humanos. Los modelos de estimación actuales, se basan en métricas disponibles recién en la fase de diseño temprano del sistema. Los gerentes deben elegir entre tomar decisiones tempranas bajo alta incertidumbre, o mitigarla pagando con tiempo y encareciendo el proyecto [19].

Putnam y Myers [20] definen 5 métricas medulares para el éxito de la gestión de proyectos de software a) cantidad de funcionalidad, b) productividad, c) tiempo, d) esfuerzo y e) confiabilidad. Las primeras cuatro corresponden al proceso de desarrollo y la quinta a la explotación del sistema y se encuentra fuera del alcance de este trabajo.

Nuestra investigación apunta a desarrollar modelos formales (automatizables) de estimación del tiempo y esfuerzo de desarrollo de sistemas de información, basados en bases de datos relacionales, con procesos evolutivos y ágiles y generación automática de código a partir de especificaciones formales[1].

Estos modelos emplean como parámetros de entrada la eficiencia del grupo de desarrollo, la volatilidad de los requerimientos, la rapidez de ejecución del projecto y la complejidad del sistema a desarrollar **medida con independencia de la tecnología** a utilizar.

En este trabajo se presenta un análisis causal de los factores relacionados con la complejidad cognitiva de un SG y el tiempo y esfuerzo requeridos para su desarrollo. Sobre la base de este análisis se discuten métricas de complejidad cognitiva para SG que consideran las contribuciones que a la misma realizan los datos y los procesos, independientemente de la tecnología usada para su desarrollo y se presentan modelos de estimación de tiempo y esfuerzo de desarrollo en base a elementos disponibles en etapas muy tempranas del proceso. Las métricas usadas son calculadas a partir de información obtenida automáticamente desde las especificaciones del sistema a construir. Para ello se usó una herramienta de recolección automática de métricas que fue explicada en un trabajo previo[2] [22].

2 TRABAJOS PREVIOS

Es bien conocido el trabajo de Albrecht [1] sobre puntos funcionales y todas sus evoluciones y variantes hasta llegar al Standard ISO/IEC 19761:2003 [9].

Grompone [14] propuso una herramienta para apoyar el cálculo de los puntos funcionales y, a partir de los puntos funcionales del sistema a construir, una curva empírica para estimación de esfuerzo de proyectos desarrollados con la herramienta de especificación formal usada para desarrollar los proyectos que hemos relevado.

---

[1] Nuestro criterio respecto de la formalidad de la especificación no está restringido al rigor matemático de los lenguajes de especificación formal. Consideramos que si la especificación contiene información suficiente para generar automáticamente la aplicación en cualquier lenguaje y plataforma (siempre que esté disponible el generador correspondiente) y esta especificación es publicada de forma que pueda automatizarse su procesamiento, entonces es una especificación formal. Seguimos el pensamiento de Briand [8] en el sentido de que en el estado del arte actual de la ingeniería de software, apegarse a un formalismo exagerado es cerrar las puertas a la adquisición de nuevos conocimientos.

[2] Esta herramienta fue desarrollada por grupos de estudiantes que realizaban su trabajo final de grado en el Laboratorio de Investigación en Sistemas de Información de la Universidad ORT Uruguay con el apoyo del Departamento de Informática del Ministerio de Transporte y Obras Públicas de Uruguay.

Chalar, Dávila y Berriel [12] encontraron una notable disparidad de criterios para evaluar la complejidad y dificultad de construcción de los distintos objetos que componen un sistema en una encuesta realizada entre expertos y propusieron investigar unidades que fueran útiles como patrón de medida de tamaño de aplicaciones.
Sin embargo todas estas propuestas presentan la misma debilidad: **no son automatizables y dependen del juicio experto**. Putnam [21], Boehm[4, 5] y en general todos los autores que han desarrollado y publicado modelos de estimación de tiempo y esfuerzo de desarrollo, encontraron que existe una relación exponencial entre las variables de entrada y el esfuerzo o tiempo [14].
Salvetto y Nogueira [22] presentaron una herramienta de recolección automática de métricas a partir de especificaciones formales que fue usada para la recolección de datos para este trabajo y definieron la métrica KBIS (Knowledge Base Intelectual Size).
Shao y Wang [24], propusieron una interesante métrica de complejidad cognitiva del software, basada en las estructuras de control y su anidamiento. Latorres y Salvetto [17] desarrollaron en el Ministerio de Transporte y Obras Públicas de Uruguay un IDE de ciclo completo que permite tomar automáticamente y en tiempo real las métricas del Personal Software Process (PSP) [15,16] y relacionar los trabajos con requerimientos posibilitando la medición directa de la volatilidad de los mismos y la cuantificación de su influencia en el esfuerzo y tiempo de desarrollo. No obstante, todavía no existe una muestra importante de proyectos desarrollados con esta herramienta por lo que en este trabajo la volatilidad de los requerimientos ha sido estimada por los gerentes de los proyectos y no medida directamente Bennet [2] publicó un modelo de estimación temprana del esfuerzo de desarrollo de sistemas siguiendo la metodología del DoD tomando métricas desde diagramas idef0 e idef1x. En su trabajo, destaca que no pudo encontrar ninguna correlación lineal entre las variables de entrada y tuvo que recurrir a normalizarlas mediante logaritmos. Esto es coincidente con lo encontrado por nosotros durante el análisis exploratorio de datos y la mayoría de los modelos de estimación que muestran relaciones funcionales exponenciales.

3 NUESTRAS HIPÓTESIS DE TRABAJO

1) Si sólo se desarrolla funcionalidad no redundante y de valor para el negocio o para algún usuario**, la complejidad esencial del sistema a desarrollar queda determinada por la integración de las visiones de datos de los usuarios del sistema y esta puede medirse sobre la base del esquema de bases de datos relacionales generado a partir de la integración de las visiones de datos de los usuarios.**
2) El esquema de base de datos relacional así obtenido contiene información suficiente como para ser usada como entrada de modelos de estimación de tiempo y esfuerzo de desarrollo.
3) Aunque no comprendamos en profundidad los complejos procesos internos e interacciones que ocurren durante el proceso de desarrollo, podemos observarlo y construir modelos que estimen aceptablemente su resultado global. En particular, si el desarrollo se realiza usando herramientas que automaticen la generación del código, la integración de módulos y apoyen el trabajo de grupos reducidos con usuarios integrados a los mismos usando una metodología estándar, se estará reduciendo las fuentes de variabilidad y será posible encontrar modelos útiles para la estimación temprana.

4 CARACTERÍSTICAS DE LA MUESTRA

Los proyectos fueron desarrollados con metodologías ágiles, ciclo de vida evolutivo y herramientas de especificación formal. Se trabajó con equipos de desarrollo reducidos (2 a 5 personas) con usuarios integrados a los mismos. Todos los sistemas fueron desarrollados a partir de las vistas de datos de los usuarios y sus solicitudes. Se desarrollaron utilizando tecnología de bases de datos relacionales. Los sistemas estudiados son Sistemas de Información del tipo data-strong [13] donde los algoritmos de alta complejidad o no existen o son excepcionales En todos los casos, las métricas de entrada de nuestros modelos fueron tomadas en forma automática, con la herramienta de que disponemos, a partir de las especificaciones.
Las únicas excepciones fueron la volatilidad de los requerimientos y el tiempo y esfuerzo requeridos para el desarrollo del proyecto que fueron estimados por los gerentes de proyecto. En relación a estas dos últimas variables, se dispuso de 8 proyectos, muy bien medidos ya que existían registros detallados de horas de trabajo. La duración y esfuerzo de los proyectos restantes fueron estimadas por los gerentes en el transcurso de entrevistas en las que se completaron formularios describiendo las características de los mismos. Puede consultarse los datos relevados en la tabla 4.

La herramienta de especificación formal usada para el desarrollo tiene la capacidad de automatizar[3] el trabajo con los usuarios en el ámbito de sus vistas de datos no normalizadas de la realidad, la integración de estas vistas de datos de usuario, la generación del esquema relacional que representa la integración de las vistas de datos de usuario, la determinación del impacto de un cambio en el esquema relacional, la generación de los programas necesarios para migrar los datos cuando se produce un cambio en las vistas de datos de usuario que impacta en el esquema relacional, la generación del código en diversos lenguajes y plataformas a partir de la especificación (independizándonos por tanto de la tecnología), la integración de varias especificaciones en una sola, detectando las posibles inconsistencias (que obviamente tendrán que levantar los desarrolladores), la publicación de la

[3] La herramienta utilizada en esta investigación fue GeneXus de Artech, www.artech.com

especificación posibilitando la automatización de la obtención de métricas y el manejo de un nombre único para cada atributo.
Se analizaron proyectos desarrollados con una misma metodología de trabajo estable y equipos de desarrollo de similares características, por lo que **la eficiencia es constante dentro de la muestra**.

5 LA METODOLOGÍA Y LA HERRAMIENTA DE ESPECIFICACIÓN

La herramienta de especificación genera el diseño de la base de datos relacional, el código para la creación de la base de datos y los programas aplicativos requeridos por el sistema de información en construcción a partir de su especificación formal[4]. También es posible exportar la especificación por medio de un archivo XML y acceder a los detalles de la especificación en forma tabular.
La herramienta almacena información en una base de conocimiento (KB). En este artículo, siguiendo la terminología estándar de Genexus, KB (Knowledge Base) refiere a un conjunto de objetos que contiene la especificación del sistema en un lenguaje propietario. Estos datos son de gran utilidad para capturar métricas automáticamente.
La metodología asociada a la herramienta se apoya en dos premisas. Primero en una organización mediana o grande, **nadie tiene una visión global de los datos y los procesos**. Por tanto, se requiere integrar las visiones de diferentes usuarios. Segundo, **los requerimientos y las estructuras de bases de datos sufren cambios continuos e inevitables a lo largo del tiempo.**
La especificación permite modelar el sistema, generar la base de datos en tercera forma normal y el código para la aplicación en el lenguaje y plataforma de ejecución deseados. Cuando se produce un cambio en los requerimientos, se ajusta la especificación y la herramienta genera el nuevo esquema de base de datos, reorganiza la base de datos y genera el código requerido.
La especificación usa diferentes tipos de objetos: vistas de datos de usuarios, procedimientos, reportes, paneles de trabajo, y menús.
La construcción de estos objetos se basa en atributos, reglas, eventos, subrutinas y programas externos.
Las Vistas de datos de Usuario modelan las entidades del mundo real según las perciben los usuarios finales. La herramienta diseña y genera la base de datos a partir de estos objetos de especificación. Este es un factor muy importante para nuestra investigación, ya que al generarse el esquema relacional automáticamente a partir de las vistas de usuario y siguiendo un modelo matemático, podemos medir la complejidad esencial directamente a partir de él sin que se produzcan desviaciones atribuibles a estilos de trabajo en el diseño de bases de datos por parte de los desarrolladores o a la forma como los usuarios ven el mundo
Los **Procedimientos** modelan procesos por lotes para gestión de las bases de datos.
Los **Reportes** permiten la recuperación de información.
Los **Paneles de trabajo** representan consultas interactivas a la base de datos. Finalmente, los **menús** organizan los objetos anteriores. Puede encontrarse mayor información sobre esta herramienta y nuestra línea de investigación principal en [22] .

5.1 Publicación de la especificación

La herramienta publica los detalles de la especificación en dos sabores: textual en XML y tabular por medio de un OLEDB provider. El primero apunta a exportar la KB y el segundo permite a los programadores acceder a los metadatos y extender y adecuar la herramienta a sus necesidades. El OLEDB provider expone los artefactos de especificación y sus relaciones. Esta facilidad fue un factor decisivo en la elección del tipo de herramienta.

5.2 Acceso a las fuentes de datos sobre la complejidad del sistema

Hay tres fuentes de medición de la complejidad, todas ellas derivadas de la especificación del sistema sin intervención humana y por lo tanto no sesgadas y adecuadas para ser recolectadas automáticamente e incorporadas a un ambiente integrado de desarrollo: a) artefactos de especificación y sus relaciones b) especificaciones textuales del sistema c) código fuente generado.
Estas fuentes de información se encuentran disponibles en todas las etapas del ciclo de vida. Obviamente la generación de código tiene algunas limitaciones en etapas tempranas.
Las dos primeras dependen solamente de la especificación, en cambio la tercera es dependiente del lenguaje generado.
Tratando de aprovechar la facilidad de acceso a los artefactos de especificación y sus relaciones proporcionada por el OLEDB provider, las métricas medulares de nuestra investigación fueron calculadas a partir de la exposición tabular de los artefactos de especificación y sus relaciones.

---

[4] Puede obtenerse más información sobre GeneXus en http://genexus.com. Trabaja con especificaciones formales en el sentido que a partir de ellas puede generar la aplicación en cualquier lenguaje y plataforma si se dispone del generador correspondiente.

6 DETERMINACIÓN DE MÉTRICAS CANDIDATAS.

6.1 Preguntas de investigación de este artículo

En este artículo apuntamos a responder las siguientes preguntas de investigación:

- ¿Cuáles son las métricas tempranas que permiten predecir tiempo y esfuerzo de desarrollo?
- ¿Es posible considerar una métrica de complejidad esencial de un sistema de información independiente de la tecnología usada para su implantación?
- ¿Se puede generalizar esta métrica a sistemas desarrollados con otras herramientas, especialmente cuando estas no tienen la capacidad de generación automática del esquema a partir de las vistas de datos de usuario?

6.2 Características de las métricas buscadas

El uso de especificaciones formales es vital porque: a) nos permite usar las mismas hipótesis que [19]; b) posibilita una recolección de métricas no ambigua y automática; c) permite la medida temprana de la complejidad basada en los requerimientos; y d) posibilita una medición post mortem objetiva y automática del tamaño de un proyecto midiendo la complejidad total de la especificación final.

Antes de seleccionar las métricas, establecimos las características que debían tener que incluyen aspectos prácticos, precisión, disponibilidad, factibilidad de automatización, resistencia del personal y significación para el usuario.

A continuación presentamos un resumen de las principales características encontradas:

Disponibilidad temprana: necesitamos métricas disponibles en estados muy tempranos del proceso de desarrollo.

Tardías: también necesitamos métricas tardías porque usamos métricas post mortem para encontrar correlaciones con métricas tempranas. Los modelos post-mortem no se presentan en este artículo por limitaciones de espacio. Pueden encontrarse en [23].

Repetibles: necesitamos que diferentes observadores arriben al mismo resultado.

Recolectables automáticamente: la recolección de métricas manuales es muy pesada, consume tiempo, está sujeta a desviaciones y es inexacta.

Fácil de calcular: queremos métricas calculables con algoritmos simples.

No invasivas: la recolección de métricas no debe interferir con el proceso de desarrollo.

Claro significado: los usuarios deben entender el significado de la métrica.

Simplicidad: preferimos métricas con el menor número de parámetros siguiendo el principio de la navaja de Occam.

6.3 Independencia de la Tecnología

Buscamos una medida de complejidad de una base de conocimiento, que debe ser independiente de la tecnología usada. Esta independencia es consecuencia inmediata del hecho de que, a partir de la especificación formal, es posible generar la aplicación en diferentes plataformas y lenguajes.

6.4 Complejidad

Necesitamos medidas objetivas, repetibles y automáticas para prescindir de las interpretaciones de los expertos que pueden ser sesgadas [19][21].

A partir de las vistas de datos de usuario de un sistema queda determinado el esquema de bases de datos relacionales que representa la información esencial contenida en ellas, con independencia de cuales sean las vistas de datos de usuario y sus posibles intersecciones. Este esquema es generado automáticamente por la herramienta usada para el desarrollo del sistema y el relevamiento.

6.5 Métricas de complejidad derivadas de la especificación

Deseamos una métrica que esté relacionada con el esfuerzo intelectual y cognitivo requerido para construir el producto. Los modelos post-mortem no se presentan en este artículo por limitaciones de espacio [23].

Necesitamos medir la complejidad cognitiva del producto. La métrica debe depender directamente de los requerimientos funcionales en lugar de los detalles de implementación que son resueltos por el generador de código. Las distintas visiones del sistema que tienen los usuarios finales son, en cierto sentido, derivadas de detalles de implementación originados en la estructura organizacional. Por este motivo, no las tomamos en cuenta y nos basamos en el esquema automáticamente generado a partir de ellas. Por esa razón usamos algunos de los artefactos provistos en la especificación formal independientes del generador y de las visiones de los usuarios que representan problemas esenciales en el desarrollo de un sistema de información.

6.6 Métricas de complejidad a partir de los objetos de la especificación

Las **Visiones de Usuario (UV)** son los principales contribuyentes a la complejidad y aparecen durante **etapas muy tempranas de la fase de análisis de requerimientos.** Representan las visiones de diferentes usuarios a partir de las cuales la herramienta genera la base de datos. Durante algún tiempo pensamos que a partir de ellas se podría construir modelos como los que estamos buscando. Un análisis más profundo nos mostró que las UV son portadoras

de la complejidad esencial del sistema pero que esta debe ser medida indirectamente desde el esquema relacional generado a partir de ellas ya que el mismo sistema puede ser visto de diferente forma por distintos usuarios.

6.6.1 Métricas de complejidad con un enfoque cognitivo

La especificación del sistema representa su complejidad esencial, ya que a partir de ella es posible generar la aplicación en distintos lenguajes y plataformas.

Los sistemas que estamos estudiando son del tipo data-strong según DeMarco [13]. De Marco propone que estos sistemas sean estimados en base a los datos.

Existen dos fuentes principales de complejidad cognitiva los datos (Data Structure´s Cognitive Complexity) y las funciones y procesos (Process Structure´s Cognitive Complexity).

Los datos contribuyen a la complejidad mediante su estructura. Calero, Piattini, et al [11] definen métricas para evaluación de complejidad de bases de datos relacionales que clasifican en (1) **table oriented metrics** Number of unique Attributes (NA) y Referential Degree (RD) y (2) **schema oriented metrics**, Depth Referencial Tree (DRT) y Normality Ratio. El RD es el número de claves foráneas en el esquema. Nosotros agregamos el número de tablas (NT).

El DRT es la longitud del mayor camino referencial en el esquema. Los ciclos sólo se consideran una vez. La proporción de normalidad es el número de tablas en tercera forma normal dividido por el número total de tablas en el esquema. Esta métrica no es útil para nosotros ya que el esquema es generado automáticamente en tercera forma normal por lo que su valor siempre es 1. Calero, Piattini el al [11] estudian las propiedades formales que cumplen estas métricas y plantean que se debe seguir investigando su validez empírica. En este trabajo, mediante regresión lineal de las variables antes mencionadas normalizadas con transformación logarítmica, se estudian estas dependencias. Shao y Wang [24] proponen una métrica de complejidad basada en pesos cognitivos, útil para medir la contribución de la PSCC a la KBCC. Sin embargo, esta métrica se basa en las estructuras de control, elementos que se encuentran disponibles en etapas finales del proceso de desarrollo, por lo que serán consideradas con la finalidad de construir modelos post mortem para investigar la contribución de los elementos tardíos a la complejidad y poder identificar (en trabajos posteriores) un patrón de peso de una base de conocimiento, que permita gestionar los hitos y contratos de un proyecto a partir de la funcionalidad entregada y el avance en el desarrollo del proyecto.

7 MODELOS DE PREDICCIÓN DEL TIEMPO Y ESFUERZO DE DESARROLLO

Si bien, nuestro relevamiento es sobre proyectos post-mortem[5], deseamos estudiar la posibilidad de construir modelos predictores basados en elementos presentes en etapas muy tempranas del proceso. Por este motivo, no se considerarán todas las métricas disponibles. Identificamos dos fuentes principales de complejidad, una proveniente de los procesos (Process´Structure Cognitive Complexity) y otra de los datos (Data Structure´Cognitve Complexity). El PSCC no es tomado en cuenta en este trabajo porque durante el análisis exploratorio de los datos se constató que su contribución a la explicación del esfuerzo y el tiempo es poco significativa y además, se basa en métricas post-mortem. Como el esquema de datos deriva automáticamente de las vistas de datos de los usuarios y además no existen algoritmos de alta complejidad ni se han realizado trabajos que no tengan valor para el usuario, el DSCC es un elemento muy significativo para medir la complejidad esencial del sistema y explicar el esfuerzo y tiempo invertidos en su desarrollo. La volatilidad de los requerimientos, tiene una importancia muy grande porque si es muy alta puede poner el proyecto fuera de control, pero además, como el relevamiento realizado fue sobre proyectos post-mortem, contribuye a la explicación del trabajo total realizado, del cual no quedan evidencias en la KB final. En este relevamiento, la volatilidad de los requerimientos (RV) fue estimada por los gerentes de los proyectos. El esfuerzo lo medimos en meses hombre considerando los desarrolladores y los usuarios integrados al equipo de desarrollo. No se tomaron en cuenta tareas de apoyo realizadas por los usuarios como obtención de datos de prueba, datos faltantes en sistemas anteriores, etc. El tiempo fue medido en meses. Dos factores que influyen sobre el tiempo y esfuerzo son la rapidez de ejecución del proyecto y la volatilidad de los requerimientos. La rapidez de ejecución fue evaluada como esfuerzo por tiempo. Como se desea usar métricas presentes tempranamente en el ciclo de vida se utilizó esfuerzo medio del 10% inicial de ejecución del proyecto (IME). La volatilidad de los requerimientos (RV) se estimó como un valor porcentual de acuerdo a la fórmula de Nogueira [19]

$$RV=\frac{DR+BR}{NR}*100$$

Donde DR= número de requerimientos descartados (dead), BR=número de requerimientos nacidos (born) y NR = número total de requerimientos. Como ninguno de los proyectos disponía de este registro, durante las entrevistas se solicitó a los gerentes de los proyectos que la estimaran en <10% baja, 10-20% media y mayor al 20% alta, asignándose los valores 10, 20 y 30 respectivamente

Durante el análisis exploratorio de los datos no se encontraron correlaciones lineales importantes entre las variables estudiadas. Sólo se pudieron encontrar correlaciones lineales entre los logaritmos de dichas variables, lo que es coincidente con lo encontrado por otros autores [2],[4],[5],[14],[21].

[5] En la tabla 4 se presentan los datos relevados.

7.1 Construcción validación y Evaluación de los modelos

Los modelos se construyeron mediante regresión lineal sobre el 70% de la muestra elegido aleatoriamente y se validaron estudiando su valor predictivo para el total de la muestra. Se evaluó el desempeño estadístico de los modelos mediante el coeficiente de determinación (R^2), la precisión y consistencia se midió mediante el valor promedio del error relativo (MRE) y la capacidad de predicción a un determinado nivel k (PRED(k)) definidas por Conte et al [10].

El error relativo (RE) , el MRE (medium relative error) y PRED(K) se calculan

$$RE = \frac{\text{Valor Observado-Valor Estimado}}{\text{Valor Observado}} \qquad MRE = \frac{\sum |RE|}{N} \qquad \text{PRED(K)} = \frac{\text{Número de Predicciones con } |RE|<K}{\text{Número total de observaciones}}$$

En todos los casos las variables dependientes e independientes se normalizaron tomando logaritmos, pero los errores se estudiaron luego de deshacer el cambio de variable, es decir que se estudió el verdadero valor predictivo de los modelos respecto de las variables que se desea estimar y no de su logaritmo que podría ser mucho mayor. Luego los resultados se evaluaron de acuerdo a la siguiente tabla tomada de [2].

Calificación	Consistencia	Precisión
Excelente	**MRE ≤ 0.20**	**PRED(20) ≥ 0.80**
Bueno	**MRE ≤0.25**	**PRED(25) ≥0.75**
Aceptable	**MRE ≤0.30**	**PRED(30) ≥0.70**
Pobre	**MRE > 0.30**	**PRED(30) < 0.70**

Tabla 1 Escalas de calidad de modelos de estimación[2]

7.2 Modelos

7.2.1 Variables Independientes

Se construyeron **modelos tempranos** en los cuales las variables independientes fueron DRT, RD, NA, NT, IME y RV. A partir de esos modelos se definió una nueva variable independiente que es una función de DRT, RD, NA y NT que representa el peso de la complejidad cognitiva de la estructura de datos que denominamos DCXW. Luego construimos modelos a partir de esta métrica e IME y RV para investigar nuestra hipótesis de que **la estructura de los datos contiene información suficiente para definir la complejidad esencial del sistema a construir**.

7.2.2 Modelos de estimación muy temprana del tiempo

Aplicando regresión lineal sobre los logaritmos de DRT, RD, NA, NT, IME y RV, se construyó un modelo predictor del logaritmo de TIME y se llegó a la siguiente expresión:

$$\text{Ln(TIME)} = \alpha \text{Ln(DRT)} + \beta \text{Ln(RD)} + \chi \text{Ln(NA)} + \delta \text{Ln(NT)} + \gamma \text{Ln(IME)} + \lambda \text{Ln(RV)} - 4{,}252472984$$

Deshaciendo el cambio de variable

$$\text{TIME} = 0{,}014 \text{DRT}^{\alpha} \text{RD}^{\beta} \text{NA}^{\chi} \text{NT}^{\delta} \text{IME}^{\gamma} \text{RV}^{\lambda} \qquad \text{Very Early Time Estimation Model 1 (VETEM1)}$$

Donde

α = -2,41, β=- 0,33, χ = 1,82, δ=-0,14, γ=- 0,23 y λ= 0,32

Los factores DRT, RD, NA y NT representan la complejidad esencial del sistema, mientras que los restantes el efecto de la velocidad de ejecución del proyecto, la volatilidad de los requerimientos y la constante factores ambientales y del equipo de desarrollo.

Este modelo tiene R^2= 0,9698, MRE=0,0729, PRED(5)=0,5, PRED(10)=0,8, PRED(15)=0,9, PRED(20)=0,95 y PRED(25)=1 lo que muestra que predice con un error menor al 25% en todos los casos observados y tiene calificación excelente según los criterios de la tabla 1. Teniendo en cuenta la precisión con que han sido estimados los datos de entrada y a los efectos prácticos consideraremos los valores de los coeficientes de regresión redondeados a dos dígitos decimales.

Basándonos en este modelo definimos el peso de la complejidad cognitiva de los datos DCXW

$$\text{DCXW} = \text{DRT}^{-2{,}4}\ \text{RD}^{-0{,}3} \text{NA}^{1{,}8} \text{NT}^{-0{,}15}$$

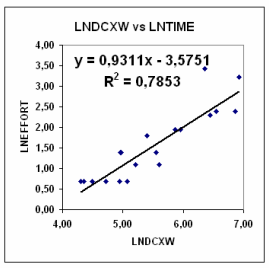
A partir de esta nueva variable se obtuvo el siguiente modelo.

$$\text{TIME} = 0{,}02\ \text{DCXW}^{1{,}08}\ \text{IME}^{-0{,}7} \text{RV}^{0{,}15} \qquad \text{Very Early Time Estimation Model 2 (VETEM2)}$$

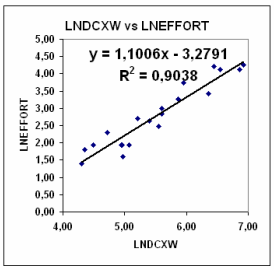
7.2.3 Resumen de modelos de estimación temprana de tiempo de desarrollo

En las Fig 1 y 2 se grafican los valores resultantes de los modelos vs los observados. Se presentan la recta y=x y la recta de tendencia del modelo así como el R^2 de la recta. También se grafican los errores relativos cometidos con la finalidad de apreciar si se trata de modelos optimistas o pesimistas y los valores absolutos de los errores relativos para evaluar los errores relativos máximos que se puede estar cometiendo. Se grafica la frecuencia relativa acumulada que representa la cantidad de observaciones donde se esta cometiendo un error <=k. Esta frecuencia, para el caso de los errores relativos tomados en valor absoluto coincide con PRED(K). En la Fig 3 se grafican los residuos.

En la tabla 2 se resumen los errores en valor absoluto mostrando la distribución de las observaciones que caen en cada rango, se presenta el valor de PRED(K) y se incluyen medidas de posicionamiento y dispersión tanto para los errores relativos como para sus valores absolutos. Se observa que el primer modelo permite realizar predicciones con un error relativo menor al 25% en el peor caso. Puede considerarse que el modelo es pesimista ya que sobreestima con mayor frecuencia de lo que subestima (Fig 3 y 4). En cambio, el segundo modelo si bien tiene una dispersión mayor parece balancear mejor las sub y sobreestimaciones. El primero califica como excelente y el otro estaría en una región intermedia entre excelente y bueno ya que el MRE es adecuado a la categoría de excelente, pero el PRED(20) es 0,75 en lugar del 0,8 requerido por los criterios de la Tabla 1. Si bien el modelo 2 tiene un error porcentual máximo mayor, no debemos olvidar que estos son errores relativos. Observando las gráficas notamos que se comporta mejor en los proyectos que insumieron más tiempo, en cambio el modelo 1 parece más adecuado para proyectos de menor duración. Se identificó una métrica de complejidad de la estructura de los datos. En las siguientes gráficas se muestra la correlación que guarda con tiempo y esfuerzo.



RE Min=2%,Max=70% MRE=30%



RE Min=3%,Max=79% MRE=25%

Se observa que la correlación es entre los logaritmos lo que coincide con lo encontrado por otros autores y por nosotros mismos. Esta correlación es más marcada para el esfuerzo. Esto no resulta sorprendente, ya que el proyecto puede ser ejecutado con diferente rapidez. Los modelos que surgen de la regresión sólo sobre DCXW, tienen los errores y R^2 que se muestran en las gráficas. Esto parece confirmar nuestras hipótesis de trabajo. Los coeficientes de correlación son 0,89 para time y 0,95 para effort.

7.2.4 Modelos de estimación muy temprana del esfuerzo

Se construyeron modelos predictores del esfuerzo de la misma forma que los de tiempo. **Se mantuvo la misma definición para el DCXW que se obtuvo a partir del modelo de tiempo, lo que no contradice la hipótesis de que la complejidad esencial del sistema guarda relación con la de los datos.**

A continuación se presentan los modelos obtenidos.

$$Ln(EFFORT)=\alpha Ln(DRT) + \beta Ln(RD) + \chi Ln(NA) + \delta Ln(NT) + \gamma Ln(IME) + \lambda Ln(RV) - 4{,}252472984$$

Deshaciendo el cambio de variable

$$EFFORT=0{,}007 DRT^{\alpha}\ RD^{\beta} NA^{\chi} NT^{\delta} IME^{\gamma} RV^{\lambda}$$ Very Early Effort Estimation Model 1 (VEEEM1)

Donde

α = -2,12, β=- 0,035, χ = 1,26 δ=0,07, γ=-0,48 y λ= 0,88

También se sugiere por razones prácticas redondear estos exponentes a dos dígitos.

De la misma forma que para el tiempo, se estimo el esfuerzo a partir de la complejidad de la estructura de los datos.

$$EFFORT=0{,}01\ DCXW^{1{,}02}\ IME^{0{,}29}\ RV^{0{,}5}$$ Very Early Effort Estimation Model 2 (VEEEM2)

En las Fig 4-6 y la tabla 3 se presentan las características de estos modelos en forma análoga a como se hizo con los de tiempo.

7.2.5 Resumen de modelos de estimación temprana del esfuerzo de desarrollo

Se presentaron dos modelos que califican como excelente de acuerdo a los criterios de la tabla 1 cuya subestimación es a lo sumo del orden del 20% y la sobreestimación del 30% en los peores casos, con un MRE del orden del 10% y sin una dispersión marcada. En virtud de lo reducido de la muestra a partir de la cual se construyeron estos modelos y los buenos resultados arrojados por ambos no es posible decidir por uno u otro.

8 CONCLUSIONES

Se definió una métrica muy temprana de complejidad cognitiva de la estructura de los datos (DCXW).Se presentaron modelos de estimación muy temprana de tiempo y esfuerzo a partir de métricas recogidas automáticamente desde las vistas de datos de usuario y que no requieren de juicio experto ni intervención humana.

Tres de los cuatro modelos obtenidos califican como excelentes de acuerdo a los criterios de Bennet [2], basados en las métricas de consistencia y precisión de Conte, Dunsmore y Shen [10]. Estos modelos usan métricas propuestas por Calero, Piattini, Polo y Ruiz [11] (DRT, RD, NA) a las que nosotros agregamos el número de tablas NT. Los resultados alcanzados, confirman la hipótesis enunciada hace tanto tiempo por DeMarco [13] de que los sistemas data-strong pueden ser estimados a partir de los datos y la bondad de las métricas de complejidad para bases de datos relacionales propuestas por Calero, Piattini et al [11] y contribuyen a aportar el soporte empírico que los propios autores expresaron en su publicación que era necesario. Los resultados de esta investigación, basados en que el modelo de bases de datos relacionales se apoya en un modelo matemático, ponen de manifiesto en forma explícita la relación que guardan estas variables con la complejidad del sistema a construir y muestran que esta estimación puede realizarse muy tempranamente y sin recurrir a juicio experto. Identificamos una métrica de complejidad cognitiva del esquema relacional a partir de la cual se construyeron modelos de estimación de tiempo y esfuerzo, que incluso se comportaron mejor, para los proyectos más grandes, que los construidos directamente a partir de las variables independientes originales. Se debe tener en cuenta que esta métrica fue definida durante la construcción del modelo de estimación de tiempo y luego usada sin cambios en los modelos de estimación de esfuerzo. Se verificó que existe una fuerte correlación entre esfuerzo y la métrica DCXW. Si se consideran los modelos que surgen de la regresión lineal solamente a partir de DCXW se puede estimar esfuerzo y tiempo con errores relativos promedio del 30% y máximos del 80%. Se intentó agregar elementos que contemplaran el peso de los procesos y las estructuras de control a los modelos post mortem, pero si bien aumentó la precisión de los modelos, no se apreciaron diferencias sustanciales (esto no aparece en el trabajo por razones de espacio, puede encontrarse mayor información en [23]). Esto confirma, o al menos no contradice nuestra hipótesis de que la complejidad esencial de un SG se encuentra en las visiones de usuario y puede ser estudiada a partir del esquema de bases de datos relacional derivado de ellas. Las métricas elegidas son intuitivas (para nosotros, desafortunadamente no para los usuarios) y fueron sugeridas con anterioridad por otros autores. Las métricas usadas fueron obtenidas automáticamente. Todo esto contribuye a aumentar la confianza en los resultados obtenidos. A pesar de que las métricas surgen automáticamente de las vistas de datos de usuario, **no son intuitivas para ellos.** Como, además, el mismo sistema puede ser visto de diferentes formas por distintos usuarios, no pudimos por el momento construir modelos de estimación con métricas obtenidas directamente a partir de las vistas de datos de usuario, sin embargo algunos de estos parámetros como NA y RD pueden ser estimados. Estos modelos son de estimación muy temprana para quien disponga de una herramienta similar a la que se usó para desarrollar los sistemas estudiados. También podría calcularse la métrica DCXW si se dispusiera de una herramienta que, dadas las vistas de datos de usuario, calcule DRT, RD, NA y NT. Naturalmente, luego deberá ser relacionada con tiempo y esfuerzo para cada ambiente de desarrollo, especialmente si no se dispone de una herramienta que nos independice de la tecnología. Los modelos encontrados, pueden ser también usados para estimar el efecto de diferentes velocidades de ejecución del proyecto, volatilidades de los requerimientos y/o el agregado o supresión de vistas de datos de usuario sobre el tiempo y esfuerzo requeridos. Estos modelos pueden ser útiles para elaborar planes de versiones e iteraciones en el desarrollo y discutir con los usuarios las reales posibilidades técnicas de alcanzar determinadas fechas sobre una base objetiva. También pueden ser útiles para gestionar contratos y ajustar precios de acuerdo a las variaciones experimentadas en el esquema de datos, rapidez de ejecución y volatilidad de los requerimientos con respecto a las acordadas inicialmente. La conclusión final parece ser que si normalizamos nuestras prácticas y concentramos nuestro esfuerzo intelectual en los aspectos no automatizables del proceso podemos ser mucho más predecibles y eficientes, confirmando nuestras hipótesis de trabajo.

9 LÍNEAS DE TRABAJO FUTURAS

Desarrollar y publicar un algoritmo que, a partir de las vistas de datos de usuario, nos de las métricas usadas en este trabajo de manera de poder calcular tempranamente estos valores y que cada equipo de desarrollo pueda vincularlos con su experiencia.

Buscar una métrica que permita *pesar* la funcionalidad entregada en una KB y apoye la gestión de contratos y proyectos en base al avance y la funcionalidad entregada.

10 RECONOCIMIENTOS

Los valiosos y constructivos comentarios de los revisores anónimos contribuyeron a mejorar este artículo. Este trabajo no hubiera sido posible sin la colaboración de Karina Santo, José Luís Chalar, Gustavo Carriquiry y Claudia Araujo de Artech Consulting, Enrique Latorres y Jose Luís Subelzú del Departamento de Informática del Ministerio de Transporte y Obras Públicas de Uruguay, Juan Andrés Leiras del Departamento de Informática de Sanidad Policial, Óscar Camargo de la Universidad del Trabajo de Uruguay y la Universidad ORT Uruguay y Gonzalo Pérez y Joaquín González de CONEX Consulting. Fueron fundamentales los aportes de Nicolás Jodal, Karina Santo y José Luis Chalar de Artech Consultores. Merecen un comentario aparte el soporte, aliento, numerosas conversaciones e inteligentes sugerencias de Julio Fernández, Decano de Desarrollo Académico de la Universidad ORT Uruguay. Representaron un invalorable aporte las conversaciones mantenidas con Ernestina Menasalvas tanto en UPM como en ORT. Fueron importantes los acertados y constructivos comentarios de Luis Olsina de la Universidad Nacional de La Pampa durante sus visitas a ORT y en el transcurso de las defensas de los proyectos finales de grado de los

estudiantes que participaron de esta investigación. Estudiantes de grado y postgrado del Laboratorio de Investigación en Sistemas de Información y la Cátedra de Teoría de la Facultad de Ingeniería de la Universidad ORT Uruguay colaboraron con esta investigación. Las visitas a UPM del primer autor son financiadas por el Programa de Desarrollo Tecnológico del BID.

11 REFERENCIAS

1 Albrecht, A. J. Measurement application developments, Proceedings of IBM Applications Development Joint SHARE/GUIDE Symposium, Monterey, CA, pp 83-92,1979

2 Bennett , Warren. Predicting software system development effort Very early in the life-cycle using Idef0 and idef1x models. Phd Dissertation Submitted to the Faculty of Mississippi State University. December 1996.

3 Boehm, B. A Spiral Model of Software Development and Enhancement. Computer. May, 1988.

4 Boehm, B. et al. Software Cost Estimation with COCOMO II. Prentice Hall, 2000

5 Boehm, B. Software Engineering Economics. Prentice Hall, 1981.

6 Boehm, B. Software Risk Management. IEEE Computer Society Press. 1989.

7 Boehm, B., Madachy R., Selby, R. Cost Models for Future Software Life Cycle Processes: COCOMO 2.0. http://sunset.usc.edu/COCOMOII/cocomo.html.

8 Briand Lionel, El Emam Khaled , Morasca Sandro. On the Application of Measurement Theory in Software International Software Engineering Research Network technical report #ISERN-95-04

9 Common Software Internacional Consortium Full Function Point Measurement Manual. (THE COSMIC IMPLEMENTATION GUIDE FOR ISO/IEC 19761: 2003) VERSION 2.2 January 2003

10 Conte, Samuel D., H. E. Dunsmore, and Vincent Y. Shen. 1986. Software engineering metrics and models. Menlo Park, CA: Benjamin/Cummings.

11 Coral Calero, Mario Piattini, Macario Polo, Francisco Ruiz. Grupo ALARCOS, Departamento de Informática,Universidad de Castilla La Mancha. Métricas para la evaluación de Complejidad de Bases de Datos Relacionales. Computación y Sistemas Vol. 3, Nº 4, pp 264-273, 2000, CIC – IPN. ISSN 1405-5546.

12 Dávila Daniel and Chalar Luis, Estimación de Proyectos, Métricas y Herramientas XII Genexus International Meeting.

13 DeMarco,T. Controlling Sofiware Projects. New York: Yourdon, 1982.

14 Grompone Juan. Gestión de Proyectos de Software. La Flor de Itapebí. Olmer S.A. 1996. Montevideo, Uruguay ISBN 9974-592-05-4.

15 Humphrey Watts. Introduction to the Team Software Process. Addison Wesley 1999.

16 Humphrey, Watts. A discipline for Software Engineering. Software Engineering Institute 1995.

17 Latorres, Enrique, Salvetto, Pedro, Larre Borges, Uruguay, Nogueira, Juan C.Una herramienta de apoyo a la gestión del proceso de desarrollo de software. IX CACIC, Octubre 2003.

18 Nogueira, J.C. A Formal Estimation Model for Software Projects. International Conference on Computer Science,
Software Engineering, Information Technology, e-Business, and Applications (CSITeA 02). Foz do Iguacu,
Brasil, 2002.

19 Nogueira, J.C. A Formal Risk Assessment Model for Software Projects. Ph.D. Dissertation. Naval Postgraduate
School, 2000.

20 Putnam, L. and Myers, W. Five Core Metrics: The intelligence behind successful software mangement. Dorset House.2003

21 Putnam, L. and Myers, W. Industrial Strength Software. Effective Management Using Measurement. IEEE Computer Society Press, 1997.

22 Salvetto, P., Nogueira J.C. Size estimation for Management Information Systems Based on Early Metrics. Proceedings CSITeA03. Río de Janeiro, 2003.

23 Salvetto, Pedro, Modelos Automatizables de Estimación muy temprana de Tiempo Y Esfuerzo de Desarrollo. XIV encuentro internacional Genexus. Montevideo, Uruguay 16/5/04.Transmitida en vivo por internet. La conferencia y transparencias pueden descargarse de http://www.gxtechnical.com/main/hevviewsession.aspx?8,60,581,19%3a569

24 Shao, J. and Wang Y. A new measure of software complexity based on cognitive weights. Can. J. Elec. Comput. Eng, Vol 28,Nº 2, April 2003.

25 Valverdi, Ricardo, Bohem, Barry and Reifer Donald. COSYSMO: A Constructive Systems Engineering Cost Model Coming of Age. *Submitted for the INCOSE 2003 Symposium, June 29 – July 3 2003, Washington, DC*

Anexo 1 Figuras y Tablas

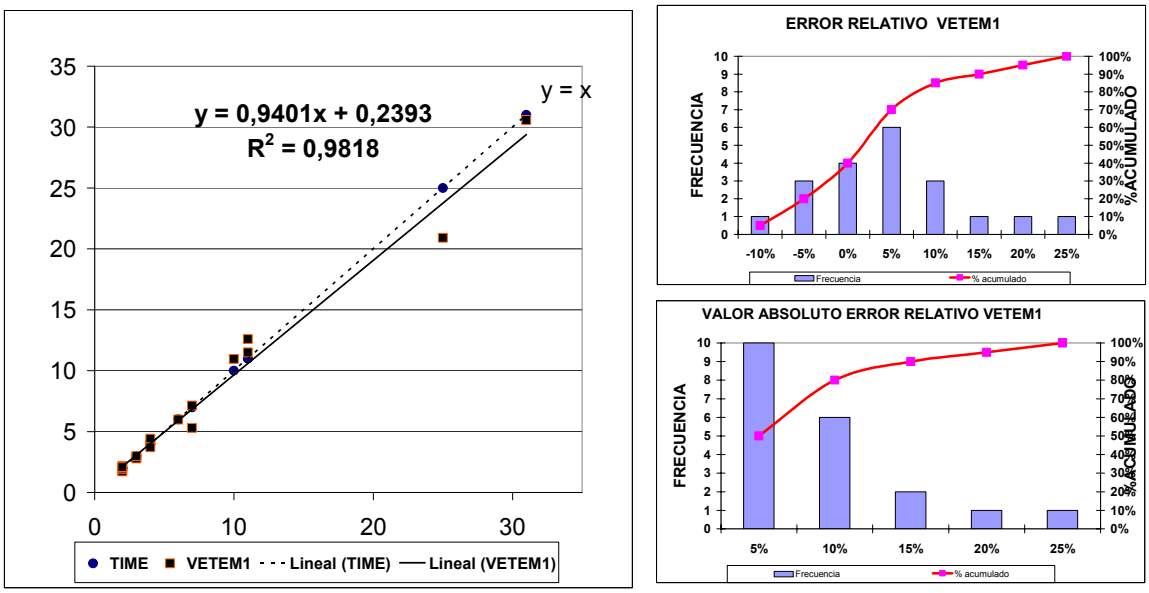


Fig 1 Modelos de Estimación Temprana de Tiempo de Desarrollo (VETEM1)

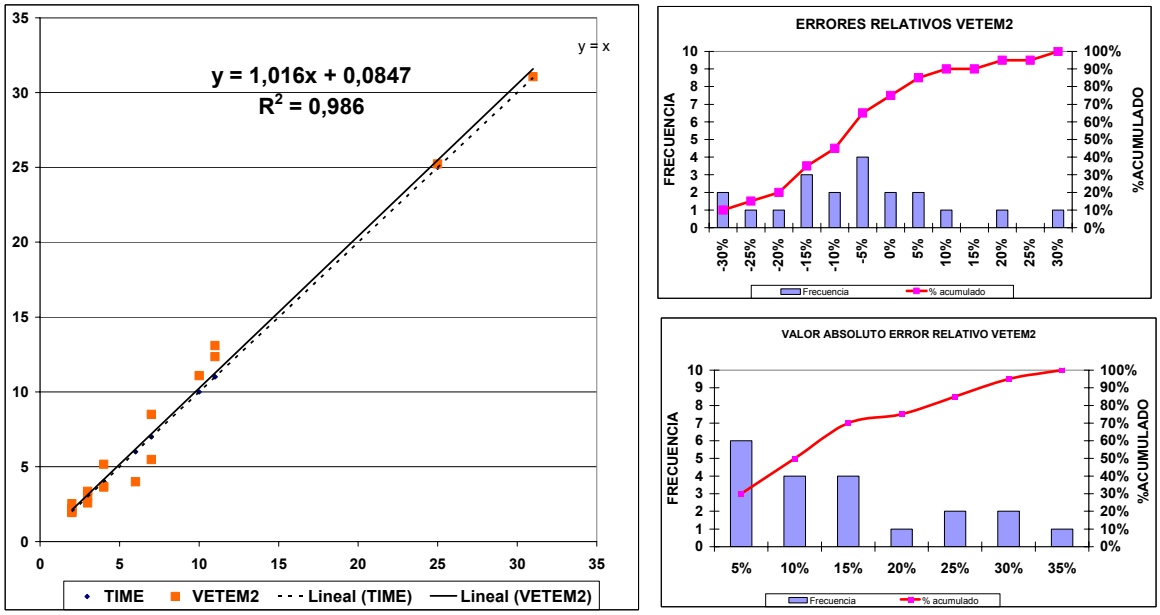


Fig 2 Modelos de Estimación Temprana de tiempo de desarrollo (VETEM2)

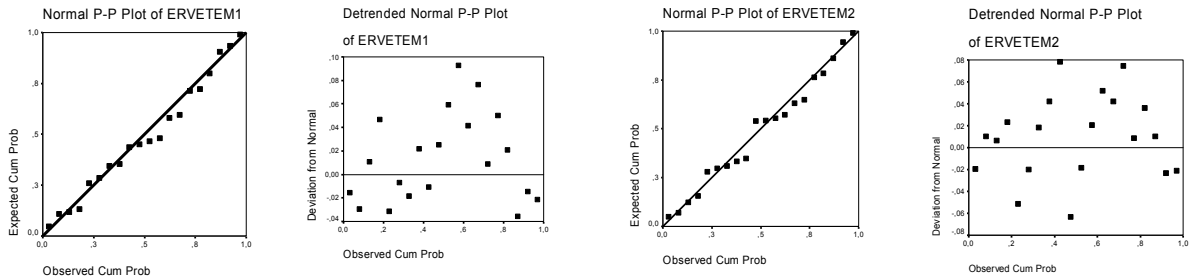


Fig 3 Gráfica de Normalidad de los Residuos de Modelos de Estimación Temprana de Tiempo

Modelos de Estimación Temprana de Tiempo																	
ERRORES RELATIVOS																	
EN VALOR ABSOLUTO																	
PRED(k)									MIN	MAX	MRE	DESV EST	MIN	MAX	PRO	DESV EST	R2	CAL
	RE	5%	10%	15%	20%	25%	30%	35%										
VETEM1	Frecuencia	10	6	2	1	1			0,3%	24,3%	7,3%	6,4%	-14,4%	24,3%	1,8%	9,6%	0,9698	Ex
	% Acumulado	0,50	0,80	0,90	0,95	1,00												
VETEM2	Frecuencia	6	4	4	1	2	2	1	0,2%	33,6%	12,2%	10,0%	-28,9%	33,6%	-3,0%	15,7%	0,9509	B
	% Acumulado	0,30	0,50	0,70	0,75	0,85	0,95	1,00										

Tabla 2 Comparación de modelos de estimación temprana del tiempo de desarrollo

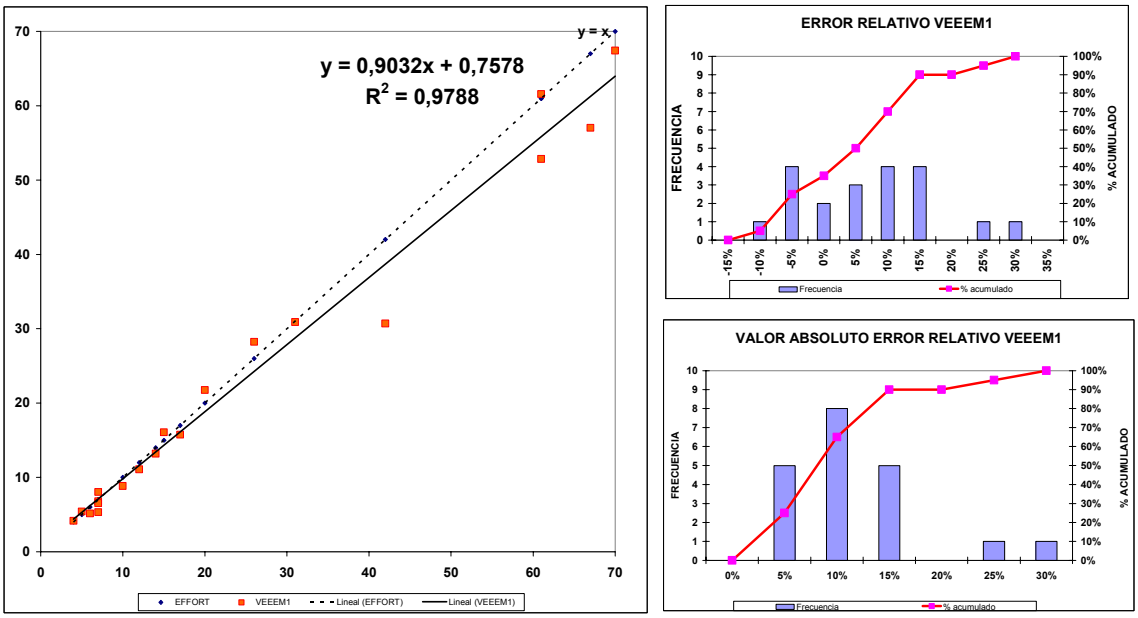


Fig 4 Modelos de Estimación Temprana del Esfuerzo de Desarrollo (VEEEM1)

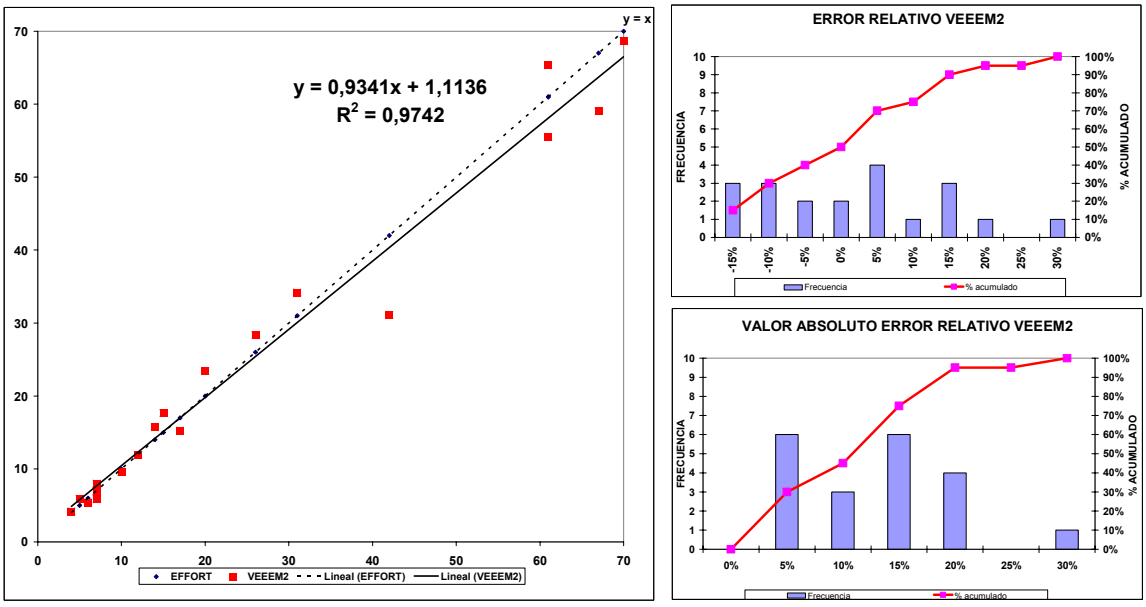


Fig 5 Modelos Tempranos de Estimación del Esfuerzo de desarrollo (VEEEM2)

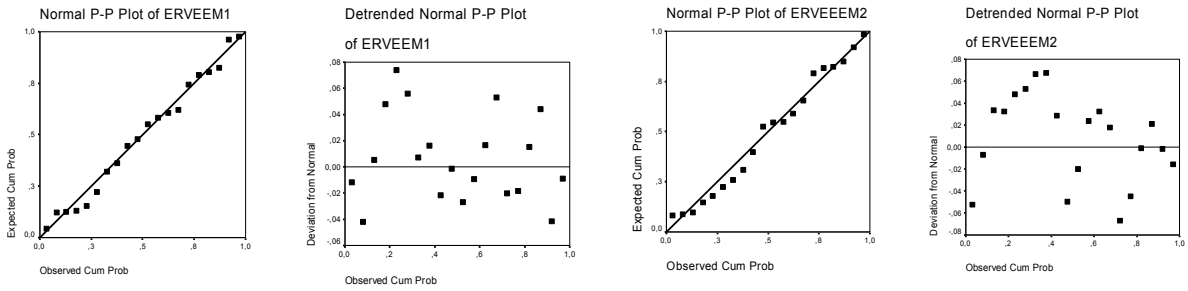


Fig 6 Gráfica de Normalidad de los Residuos de los Modelos de Estimación Temprana de Esfuerzo

Modelos de Estimación Temprana de Esfuerzo

ERRORES RELATIVOS

EN VALOR ABSOLUTO																		
	PRED(k)								MIN	MAX	MRE	DESV EST	MIN	MAX	PRO	DESV EST	R2	CAL
	RE	5%	10%	15%	20%	25%	30%	35%										
VEEEM1	Frecuencia	5	8	5	0	1	1		0,3%	26,9%	9,6%	7,0%	-15,0%	26,9%	4,3%	11,2%	0,9788	Ex
	% Acumulado	0,25	0,65	0,90	0,90	0,95	1,00											
VEEEM2	Frecuencia	6	3	6	4	0	1		0,2%	25,9%	10,1%	7,0%	-18,1%	25,9%	-0,9%	12,4%	0,9742	Ex
	% Acumulado	0,30	0,45	0,75	0,95	0,95	1,00											

Tabla 3 Comparación de modelos de estimación temprana del esfuerzo de desarrollo

num	EFFORT	TIME	IME	RV	DRT	RD	NA	NT	DCXW
1	70	25	3	20	3	114	542	85	1022
2	14	6	4	20	1	4	29	6	222
3	12	4	3	10	3	6	120	9	258
4	26	7	3	30	5	97	535	85	352
5	5	4	2	10	1	3	25	10	145
6	17	3	6	10	8	284	1042	195	270
7	15	3	6	30	5	33	259	24	183
8	42	7	6	20	6	258	842	136	387
9	4	2	2	20	4	13	123	35	73
10	20	3	8	20	6	103	489	40	271
11	7	2	4	10	5	75	329	69	160
12	67	10	5	30	6	100	948	115	627
13	7	2	3	30	5	32	182	27	90
14	7	4	2	10	1	4	23	6	143
15	61	11	5	30	4	111	588	74	693
16	10	2	4	30	4	25	136	14	113
17	61	11	6	10	4	94	666	65	956
18	6	2	2	30	3	13	79	17	77
19	7	2	4	10	3	17	111	17	140
20	31	31	1	30	1	1	44	8	577

Tabla 4 Datos Relevados

Representación visual de la Gestión de Requisitos en la Gestión de Proyectos Informáticos

Marilú Montenegro Sánchez
Universidad Carlos III de Madrid, Departamento de Informática
Av. Universidad 30, 28911 Madrid, España
marilu@inf.uc3m.es

y

Ángel García Crespo
Universidad Carlos III de Madrid, Departamento de Informática.
Av. Universidad 30, 28911 Madrid, España
acrespo@ia.uc3m.es

Resumen

Los principales objetivos del gestor de proyecto es entregar el proyecto dentro del tiempo y presupuesto con un nivel de calidad aceptable establecido en el contrato. Los proyectos informáticos están formados por un conjunto de procesos los cuales sufren continuos cambios, principalmente generados por los cambios de requisitos. El gestor es el responsable del éxito o fracaso del proyecto, por lo cual debe tratar de reducir o eliminar los factores que incrementen los costes y su duración. Se le asigna la tarea de integrar los procesos para conseguir los objetivos marcados y debe identificar las características críticas en el avance del proyecto.
Los requisitos una vez establecidos y documentados, sufren continuos cambios, en este sentido no tratamos la obtención ni el análisis de los mismos, nos centramos en su gestión, es decir, realizar el seguimiento respecto a los cambios de requisitos que se generan mientras avanza el proyecto, debido a que dichos cambios generan otros, como son: tiempo de entrega, recursos, presupuesto y calidad. Representamos el seguimiento de los cambios mediante un conjunto de gráficas y tablas, con la finalidad de que el gestor se encuentre perfectamente informado y tenga la oportunidad de tomar las medidas correctivas necesarias.
Palabras claves: Gestión de Requisitos, Gestión de Proyectos, Proyecto Software, Ingeniería del Software.

Abstract

The main objectives of project manager are to deliver the project inside time and budget with a quality level accepted established in the contract. Computer science projects are formed by a group of processes which suffer continuous changes, mainly generated by the changes of requirements. Manager is the responsible for the success or failure of the project, reason why it should try to reduce or to eliminate the factors that increase the costs and his duration. He/she is assigned the task of integrating the processes to get the marked objectives and it should identify the critical characteristics in the project development.
The requirements once established and documented, they suffer continuous changes, in this sense we do not treat the obtaining neither the analysis of the same ones, we center ourselves in their management, that is, to carry out the pursuit regarding the changes of requirements that are generated while the project development, because this changes generate other, like they are time, resources, budget and quality. We represent the pursuit of the changes by means of a set of graphic and tables with the purpose that the manager is perfectly informed and have the opportunity to take corrective actions.
Keywords: Requirements Management, Project Management, Project Software , Software Engineering.

1. Introducción

Según la IEEE se entiende por requisito [3]:

- *Una condición o capacidad necesitada por un usuario para resolver un problema o alcanzar el objetivo.*
- *Un condición o capacidad que debe cumplir o poseer un sistema o un componente del mismo para satisfacer un contrato, un estándar, una especificación, u otro documento necesitado por un usuario para resolver un problema.*
- *Una representación documentada de una condición o capacidad tal como las expresadas en 1 ó 2 anteriores*

Para [10] la Ingeniería de Requisitos involucra descubrir, documentar y mantener un conjunto de requisitos. El uso del término ingeniería implica que las técnicas sistemáticas y repetibles deberían asegurar que los requisitos del sistema sean completos y consistentes. Para [6] considera que uno de los principales objetivos de la Ingeniería de Requisitos es mejorar los sistemas de modelado y análisis con la finalidad de que las organizaciones puedan entender mejor las necesidades del cliente antes de construir el software.
Según [12] divide la disciplina de Ingeniería de Requisitos en Desarrollo de Requisitos y Gestión de Requisitos, como se muestra en la Figura 1. El Desarrollo de Requisitos involucra la elicitación, el análisis, la especificación, y validación de los requisitos del proyecto software. Esta subdisciplina abarca todas las actividades involucradas para reunir, evaluar y documentar los requisitios. La Gestión de Requisitos, incluye todas las actividades para mantener la integridad, exactitud y actualidad de los requisitos de acuerdo con el avance del proyecto.

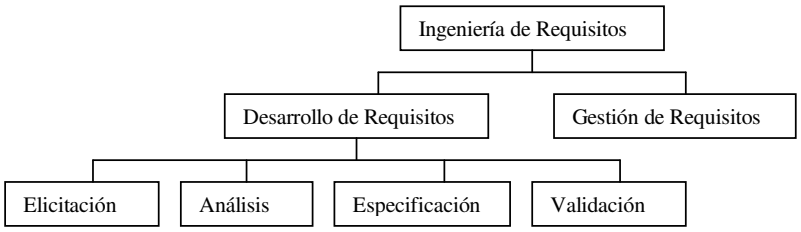


Figura 1. Subdisciplinas de la ingeniería de requisitos.

La línea base de requisitos consiste de aquellos requisitos que han sido establecidos para ser implementados bajo una primera versión. Cada versión de la especificación de requisitos tiene un único identificador para evitar confusión entre los requisitos previos y los requisitos cambiados [12].
Los requisitos se inician cuando empieza un proyecto en las etapas de análisis y especificación de requisitos, posteriormente dichos requisitos en el ciclo de vida de un proyecto pueden ser modificados por lo que se establece el concepto de Gestión de Requisitos, que es el tratamiento y control de las actualizaciones y cambios a los mismos [1], que será el centro de éste artículo. Este artículo se basa en requisitos ya establecidos y documentados, es decir, gestionar los requisitos una vez definidos. Los cambios de requisitos se pueden generar por errores o malos entendidos en el proceso de la ingeniería de requisitos.
El proyecto informático conlleva a que el producto software vive en un mundo dinámico que obliga a realizar cambios que pueden afectar a los sistemas en funcionamiento como a los que están en vías de desarrollo, si estos cambios no se gestionan de manera adecuada puede generar: que el producto no se entregue en el tiempo establecido, que se cancele el proyecto por no cumplir con los requisitos o que el cambio de requisitos consuma más recursos (persona, máquinas etc). Debido a que un proyecto informático es susceptible de cambios, habría que proceder a su actualización o a la incorporación de nuevas funcionalidades o eliminar otras, esto obliga a tener un control sobre el producto y su documentación. Es necesario gestionar estos requisitos, es decir preparar el proyecto para tratar los cambios cuando ellos se susciten y poder dirigirlos a los objetivos.
Los cambios de requisitos deben ser gestionados para asegurar que la calidad de los mismos se mantengan, los problemas suscitados por los cambios de requisitos podrían incurrir en altos costos [10]. En tal sentido podemos gestionar los requerimientos si se controla, es decir, conocer quién sugirió el cambio, cuantas veces ha sido modificado el mismo requisito, que requisitos están relacionados y cómo dichos cambios pueden influenciar en los diferentes procesos de la gestión de proyectos. En la Figura 2 presentamos estos puntos de gestión que posteriormente serán representados visualmente. Donde el control de cambios implica: propuesta de cambio, análisis de impactos, comunicación, incorporación; dentro del control de versiones tenemos: identificar las versiones de los documentos; dentro del seguimiento de los requisitos debemos: definir la relación con los otros requisitos, definir la relación con otros elementos del sistema; y, dentro del seguimiento del estado de los requisitos debemos: definir el estado de todos y cada uno de los requisitos [1],[12].
Se presenta un conjunto de gráficas con la finalidad de gestionar los requisitos en forma rápida y confiable, queremos indicar que las gráficas no representan ningún modelo matemático que nos indique exactamente que los cambios de requisitos mantengan una función estándar, lo que queremos indicar es que podemos realizar un seguimiento al dinamismo de los cambios, con la finalidad de gestionar la información de los requisitos, estar

actualizado de sus últimas versiones, conocer el impacto, para que el gestor de proyectos y los stakeholders [1] implicados tengan una mejor aproximación del mundo dinámico de los requisitos.

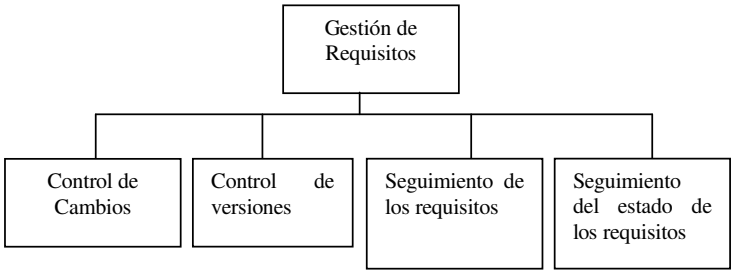


Figura 2. Principales actividades de la gestión de requisitos

2. Identificación de la Gestión de Requisitos dentro de los Estándares.

No existe ningún estándar que proporcione las orientaciones visuales para una gestión de requisitos. Sin embargo, determinados procesos de las normas utilizan el proceso de requisitos como punto de partida para iniciar un proyecto. No existe un procedimiento exacto para gestionar los requisitos, pero se debe tener en cuenta un conjunto de normas para realizar dicha actividad, que dependen de la magnitud de la organización y de la complejidad del sistema a desarrollar. Entre dichas normas se considerarán [1]:

- Especificar las normas, técnicas y herramientas para controlar las versiones
- Puesta de requisitos en la línea base
- Especificar la normas para analizar el impacto
- Especificar las normas para gestionar el origen de los cambios
- Designar al personal responsable para el cambio de requisitos
- Especificar el estado de los requisitos.
- Reflejar los cambios suscitados frente a los otros procesos, como costes, recursos humanos, plazos.

2.1 La Norma ISO/IEC 15504

Figura 3. Procesos de la dimensión proceso.

[1] Conjunto de personas que participan en el resultado del proyecto informático, como son: el jefe del proyecto, analistas, desarrolladores, cliente, usuarios etc. [9]

La norma ISO/IEC15504 proporciona un marco para evaluar el proceso software. Las organizaciones pueden utilizarla para planificar, gestionar, ejecutar, controlar y mejorar los procesos de adquisición, suministro, desarrollo, operación, evolución y soporte del software. Presenta un modelo de referencia cuya arquitectura es bidimensional: la dimensión procesos y la dimensión capacidad de procesos, con el propósito de proporcionar una base común para los diferentes modelos y métodos de evaluación de procesos software.
La dimensión procesos (Figura 3) se agrupa en tres procesos: primarios, de apoyo y organizacional. En total contiene cinco categorías: cliente-proveedor, ingeniería, soporte, gestión , y organización [4]. Dentro de la categoría cliente-proveedor se especifica algunos aspectos relacionados a la gestión de requisitos. Los procesos de ésta categoría consisten de procesos que están directamente relacionados con el cliente, y el que desarrolla el sistema, tratando la gestión de requisitos en el proceso de Elicitación de Requisitos (ER) en un sentido concreto, simple y general.

2.2 Madurez del Software

El modelo de Madurez de la Capacidad del Software (CMM), fué desarrollado por el Software Engineering Institute (SEI) con la finalidad de ayudar a las organizaciones a mejorar su proceso software mediante un conjunto de áreas clave de proceso (KPAs), donde cada KPA se compone de prácticas claves, que describen las actividades para la implementación de una capa; organizados en 5 niveles de madurez: inicial, repetible, definido, gestionado y optimizando (Figura 4). Para alcanzar un nivel de madurez, las KPAs a ese nivel deben ser satisfechas [5],[8].
Las prácticas claves se agrupan en cinco categorías que el CMM denomina características comunes. Cada KPA tiene los cinco tipos de características comunes y al menos una práctica clave bajo cada característica común. Los nombres de las características comunes son: Compromiso para realizar (Co), Capacidad para realizar (Ab), Actividades realizadas (Ac), Medición y análisis (Me) y Verificación de la implementación (Ve). Una práctica de compromiso para realizar (Co) es generalmente una política de la organización firmada por la alta dirección. Las prácticas de Capacidad para realizar (Ab) aseguran que los recursos (generalmente tiempo y dinero) estén disponibles para llevar a cabo las otras prácticas y que las condiciones de capacitación, como el entrenamiento, estén satisfechas. Las prácticas de las Actividades realizadas (Ac) sugiere las acciones que pueden realizar el equipo técnico para llevar a cabo la planificación, seguimiento o entrenamiento dentro de esa área de proceso. La característica común Medición y análisis (Me) asegura que el estado de las prácticas de la KPA se conozca cuantitativamente. Y, la característica común de Verificación de la implementación (Ve) demanda una revisión regular por la dirección para asegurar la implementacion y resolver los problemas que se susciten [8].
Según el CMM, los objetivos de la gestión de requisitos son: los requisitos del sistema se controlan a fin de establecer una línea base; y (2) los planes, productos y actividades del software se mantienen consistentes con los requisitos del sistema [8]. Además, los requisitos deben estar documentados y controlados dentro de una línea base para la gestión del proyecto, para asegurar que la planificación, los entregables y las actividades sean consistentes con los requisitos.

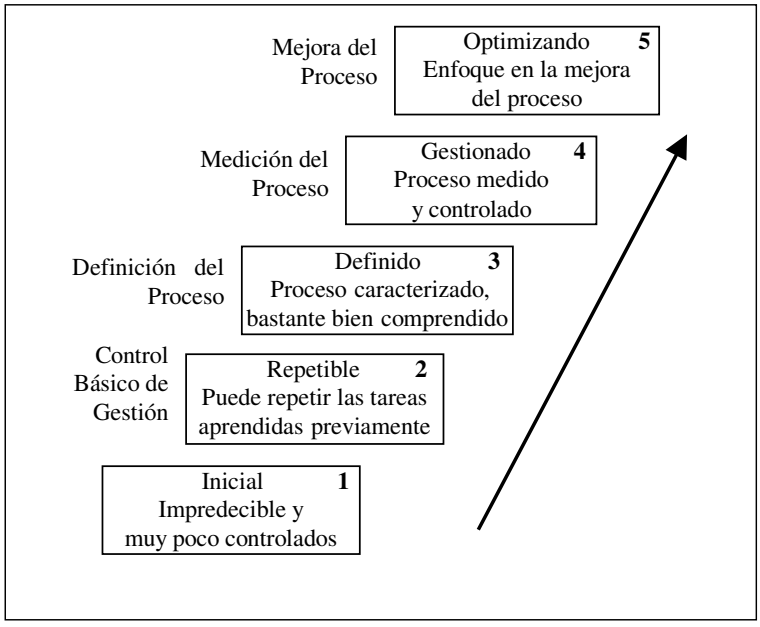


Figura 4. Evolución del proceso de la ingeniería del software

En el nivel 2 (Figura 5), Repetible se encuentra el área clave de proceso Gestión de Requisitos, que tiene 12 prácticas clave: Compromiso (1), capacidad (4), actividad (3), medición y análisis (1), verificación de la implementación (3). Es decir, realiza un enfoque más amplio de la gestión de requisitos.

Nivel de Madurez	**Área Clave de Proceso**
Nivel 2	Gestión de Requisitos Planificación del Proyecto Software Seguimiento y Control del Proyecto Software Aseguramiento de la Calidad del Software Gestión de la Configuración del Software Gestión de la Subcontratación del Software.
Nivel 3	Enfoque en el Proceso de la Organización Definición del Proceso de la Organización Programa de Entrenamiento Gestión Integrada del Software Ingeniería del Producto Software Coordinación entre Grupos Revisiones por iguales
Nivel 4	Gestión Cuantitativa del Proceso Gestión de la Calidad del Software
Nivel 5	Prevención de Defectos Gestión del Cambio de Tecnología Gestión del Cambio del Proceso.

Figura 5. Áreas claves de proceso

2.3 La Gestión de Proyectos.

El Project Management Body of Knowledge (PMBOK) es una guía estándar para la gestión de proyectos desarrollada por el Project Managemet Institute (PMI) que identifica y describe un conjunto de áreas de conocimiento y prácticas que son aplicables en la gestión de proyectos en términos de sus procesos componentes. Las nueve áreas de conocimiento son: Gestión de Integración del proyecto, Gestión del Alcance, Gestión del Plazos, Gestión de Costes, Gestión de Calidad, Gestión de Recursos Humanos, Gestión de Comunicaciones, Gestión de Riesgos y Gestión de Adquisición. La gestión de requisitos se encuentra implícita en el área de conocimiento de Gestión del Alcance cuyos procesos son: inicio, planificación del alcance, definición del alcance, verificación del alcance y control de cambios del alcance. Dentro de los procesos de inicio y planificación se elabora y documenta los requerimientos del producto a ser desarrollados, que reflejan las necesidades del cliente, se identifica los objetivos del proyecto y de los entregables, se identifica los cambios. Dentro del proceso de definición del alcance se subdivide los principales entregables del proyecto en componentes más manejables para mejorar el seguimiento de los costes, duración y los recursos estimados, facilitando una clara responsabilidad, utilizando la técnica del Work Breakdown Structure[2] (WBS). Dentro del proceso de verificación se requiere obtener la aceptación de los stakeholders, se requiere revisar los entregables para asegurar la satisfacción del cliente. Dentro del control de cambio del alcance determinamos los cambios y como influyen en los costes, en la calidad, y en la planificación [9]. Este estándar, desarrolla un conjunto de componentes para cumplir una adecuada gestión de requisitos dentro de la gestión de proyectos. En la Figura 6, resumimos la gestión del alcance de los requisitos dentro de la gestión del proyecto.

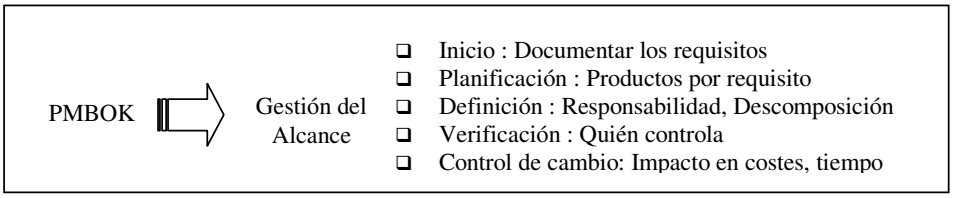


Figura 6. Identificación de la gestión de requisitos dentro de la gestión de proyectos del PMBOK

[2] Es la descomposición del proyecto en unidades de trabajo que pueden ser asignados a costes, entregables, actividades y personal con la finalidad de obtener componentes más manejables [9].

En la Tabla 1, relacionamos los parámetros establecidos para la gestión de requisitos (Figura 2) con los procesos de requisitos que se ubican en los diferentes estándares.

Gestión de Requisitos	ISO/IEC 15504 Proceso	CMM Prácticas claves	PMI Componentes
Control de Cambios	ER	Compromiso 1 Capacidad 1,2,3,4 Verificación 1,2	Inicio Definición Verificación
Control de versiones	ER	Actividad 2	Definición
Seguimiento de los requisitos	ER	Verificación 3,4	Control de cambio Planificación
Seguimiento del estado de los requisitos	ER	Capacidad 3 Actividad 1,3 Medición y Análisis 1	Control de cambio

Tabla 1. Identificación de la gestión de requisitos en los diferentes estándares.

3. Gestión de Requisitos

La evolución de requisitos es legítimo, inevitable, e incluso ventajoso. Los procesos del negocio, el mercado de oportunidades, la competencia del producto, podrían generar cambios de los requisitos y en tal sentido podría determinar la redirección del proyecto. Esto obliga a tener un control sobre el producto y su documentación para mantener la integridad y exactitud de los requisitos a medida que progresa el proyecto [12].

3.1 Control de cambio

Los cambios siempre tienen un precio, incluso un cambio rechazado consume recursos que han sido necesitados para evaluar, y decidir rechazarlo. Evaluar cada cambio se realiza con la necesidad de cumplir con los objetivos del negocio, la visión del producto y el alcance del proyecto [12].

Cada proyecto necesita incorporar los cambios más apropiados para el proyecto en un mundo controlado, en este sentido, debemos identificar quién realizó el cambio de requisito y el motivo del mismo, porque no todo el personal que pertenezca a un proyecto puede solicitar dicho cambio. Se debe asignar quiénes pueden cambiar los requisitos así como designar quién debe revisarlos o aprobarlos [8]. La tabla 2 nos indica el personal responsable para solicitar y revisar los cambios. Se debe garantizar conocer quiénes son los responsables de los cambios propuestos para controlar los cambios de una manera disciplinada.

Personal autorizado a solicitar el cambio	Personal que revisa el cambio
• Gerencia General • Otras gerencias relacionadas al Proyecto • Jefe del proyecto • Grupo de Control de Calidad • Cliente	Jefe del Proyecto
• Grupo de Ingeniería del Software Etapas : Análisis Diseño Pruebas • Usuarios	Responsable del Grupo de Ingeniería del Software

Tabla 2. Personal involucrado en el cambio de requisitos

En la Figura 7, asignamos P1, P2 y P3 a las personas involucradas a los cambios de requisitos, pudiendo encontrarse entre ellos al cliente, al grupo de ingeniería del software. Se observa que P1 ha modificado los requisitos R1 y R2 , 2 y 3 veces respectivamente, mientras que P2 ha modificado todos los requisitos, visualizando así la relación directa personal-requisitos cambiados.

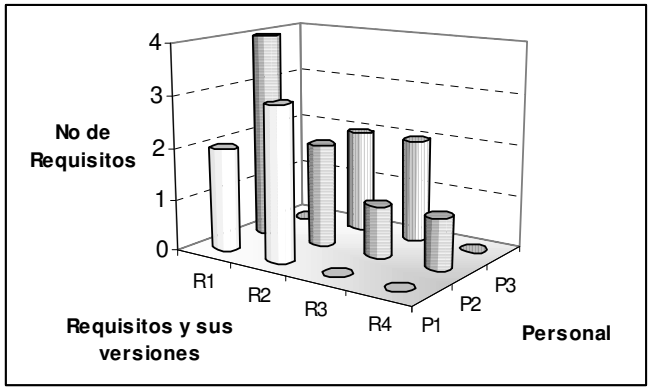


Figura 7. Personal involucrado con los requisitos y sus versiones.

La Figura 8 representa, la cantidad de requisitos cambiados por fase en el proyecto, con la finalidad de verificar posteriormente si se presenta algún cambio en la planificación del mismo. El personal P1 ha modificado en la primera Fase del proyecto 4 veces los requisitos y en la Fase 2 lo ha realizado 5 veces, debiéndose efectuar ciertos ajustes en la programación o presupuesto del proyecto por la generación de dichos cambios [12].
El Gestor de Proyecto que esté preocupado por los cambios de requisitos podría impedir los continuos cambios si conociera al responsable o a los responsables que originan dichos cambios. La Figura 9 muestra un modo de representar el número de requisitos cambiados originados por las diferentes fuentes. El gestor podría dirigirse a ellos con la finalidad de mantener una discusión provechosa acerca de las acciones que se podrían tener en cuenta para reducir el número de cambios de requisitos en el futuro [12]. Una vez que el requisito ha sido cambiado se debe comunicar a los stakeholders y a las áreas implicadas del cambio.

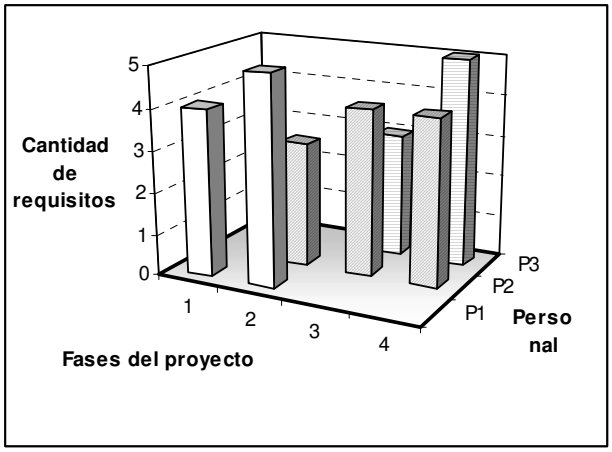


Figura 8. Requisitos cambiados por persona y por fase

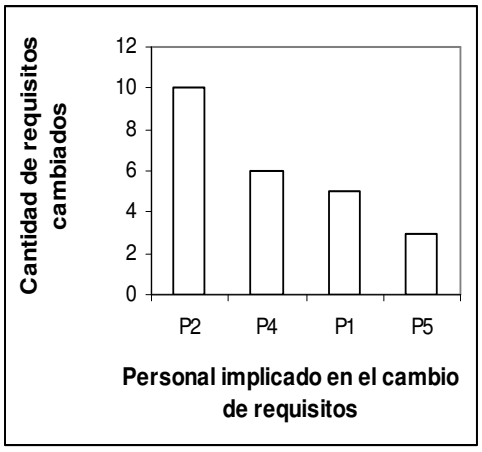


Figura 9. Origen de los cambios de requisitos

3.2 Control de las versiones

Cada requisito suele estar identificado, y como hay que reorganizar y añadir nuevos requisitos es necesario conocer la última versión de los mismos siendo de vital importancia, para evitar trabajar en requisitos no actualizados. Nos debemos asegurar que el personal responsable conozca las últimas versiones. Siendo esta actividad esencial en la gestión de requisitos, es decir:

- Las versiones de los requisitos deben estar debidamente identificadas en las líneas bases.
- El personal relacionado con el personal de requisitos debe estar actualizado con las versiones anteriores y actuales.

Las versiones de los requisitos pueden generar incremento de costes y tiempo de entrega, debemos registrar estas variaciones: fecha del cambio, persona quién lo origino, motivo del cambio y el incremento y costes que genera dicho cambio, como se puede observar éstos y otros parámetros en la Tabla 3.
Con la finalidad de observar rápidamente dichas versiones utilizamos la Figura 10, para que el gestor de proyectos o el personal relacionado con la gestión de requisitos esté actualizado con la última versión, asimismo observará en que fase dichos requisitos son cambiados. Se observa que el Requisito 3 ha sido modificado varias veces a medida

que se desarrollaba el proyecto, empieza en la línea base con la versión 1 y al llegar el proyecto a la Fase 4, el requisito se encuentra en la versión 3.5.

Código: R01			Descripción:				
Fecha	Persona	Área Funcional	Descripción del Cambio	Razón del Cambio	Versión	Incremento Tiempo	Incremento Costes
					Totales		

Tabla 3. Registro de versión de requisitos

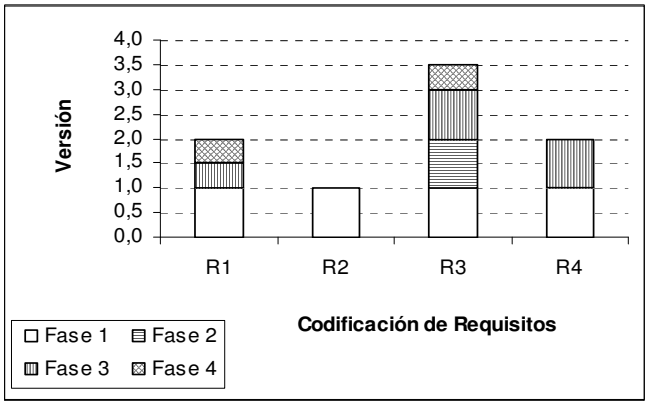


Figura 10. Versión de los requisitos por fase

3.3 Seguimiento de los requisitos

3.3.1 Impacto Frente a otros Requisitos

Algunos de los requisitos documentados están relacionados, al ser modificado uno de ellos debemos considerar el impacto que puede originar. Con la finalidad de obtener dicha influencia, se registra la información en la Tabla 4. Al cambiar el requisito R1 y R4 que están relacionados al requisito R2, éste ha sufrido una variación en el tiempo total (Tr1r2 + Tr4r2) y coste total (Cr1r2 +Cr4r2). El gestor de proyecto, estará actualizado sobre estos impactos para continuar con la correcta planificación del proyecto.

	REQUISITOS					
	R1	R2	R3	R4	R5	R6
R1		r12		r14		
R2			r23			
R3						
R4		r42	r43		r45	
…						
	Variación del tiempo y Costo					
Tiempo		Tr1r2+ Tr4r2				
Costo		Cr1r2 Cr4r2				

Tabla 4. Impacto de los requisitos relacionados

3.3.2 Definir la Relación con otros Elementos del Sistema.

El modo más común de representar los enlaces entre los requisitos y otros elementos del sistema se encuentra en la matriz de trazabilidad [12], también denominado matriz de traza de requisitos o tabla de trazabilidad, lo que significa realizar la traza del requisito desde su inicio hasta la implementación de los mismos. La forma más sencilla

de gestionar la trazabilidad es realizar un diagrama que vaya desde los requisitos elementales a los elementos del diseño, de éstos a los elementos de código, desde éstos a los casos de prueba y así sucesivamente hasta los últimos elementos [1]. Cada fila de la Tabla 5 representa un requisito específico y las columnas los diferentes elementos o productos que se deben desarrollar en las diferentes fases del proyecto. Esta matriz pretende mostrar la relación de los requisitos con los diferentes productos a medida que avanza el proyecto, con la finalidad de garantizar que los requisitos están siendo realizados. La Figura 11 nos indica el estado de desarrollo de los productos.

Requisito	Desc	Fase 1				Fase *n*		Total
		Doc	Rep	Manual	…	Doc	…..	
R1		2	5	4				
R2		1	3	3				
R3						4		
R4						5		
….								
Rn								
Productos Totales								

Tabla 5. Matriz de traza

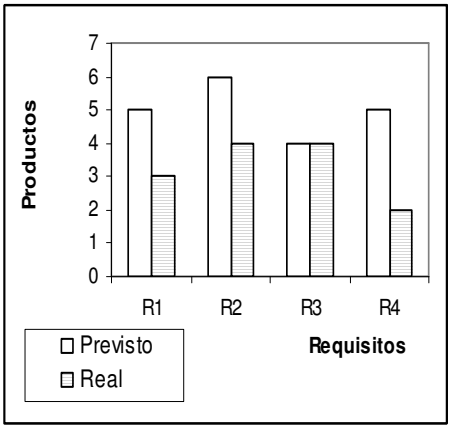


Figura 11. Validación de traza.

Se ha desarrollado muchos modelos para realizar la trazabilidad de requisitos, por ejemplo [7], presenta una herramienta denominada Sharp Trace, una propuesta de metamodelo de trazabilidad teniendo en cuenta el uso del UML. La herramienta permite definir nuevos tipos de artefactos y enlaces de trazabilidad.

3.3.3 Relaciones con otros Procesos del Sistema.

El incremento de requisitos influye en la gestión de costes y en la gestión de plazos. Verificando la información registrada en la Tabla 6. Al cambiar el requisito R1 de la Fase 1, incluirá un tiempo adicional de T1, un recurso adicional de r1, con un incremento de costes de ic1, obteniendo el impacto total por fase TF1 por cada uno de los incrementos. Queremos indicar que todo cambio de requisito no necesariamente incrementa los atributos señalados, puede suceder el caso que dichos atributos se vean disminuidos. Sólo indicamos que la variación de requisitos pueden influir en los costes, tiempo y recursos del proyecto.

Ciclos del Proyecto	Req por fase	Requisitos Cambiados	Total	Tiempo Adicional	Recurso Adicional	Incremento De Costes
Fase 1	R1			T1	r1	ic1
	R2			T2	r2	ic2
	…			……	….	…..
	TF1			$\sum T_i$	$\sum R_i$	$\sum IC_i$
Fase 2	R4			T4	r4	ic4
	R5			T5	r5	ic5
	….			……	…..	….
	TF2					
Fase n						
		Totales				

Tabla 6. Impacto del cambio de requisitos por ciclo de vida del proyecto

El incremento de costes-tiempo y el incremento de personal originado por el cambio de requisitos se observa en las Figuras 12 y 13 respectivamente.

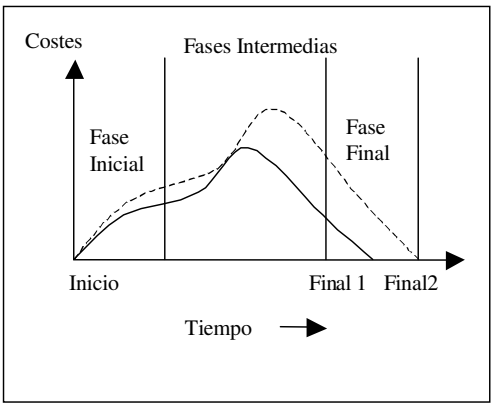


Figura 12. Incremento de costes

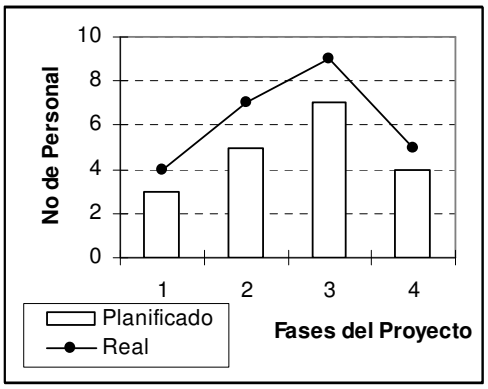


Figura 13. Incremento de personal por fase

3.4 Seguimiento del estado de los requisitos.

El estado de los requisitos es un aspecto importante en la gestión de requisitos. Se establece un conjunto de categorías para seguir a los requisitos, que pasan a formar parte de la documentación establecida como línea base. El requisito puede encontrarse en cualquiera de los siguientes estados: propuesto, aprobado, diseñado, verificado, implementado, entregado, incluyendo los posibles estados de anulado (planificado y retirado de la línea base), y rechazado (presentado, no aprobado).

De acuerdo a [5], sugiere mantener una matriz que relacione los requerimientos con el estado de los mismos, la supervisión total de un proyecto mejoraría si se pudiera mantener una información exacta y periódica de la situación y estado de los requisitos (Tabla 7).

Requisitos	Propuesto	Aprobado	Diseñado	Verificado	Implementado	Entregado	Anulado	Rechazado
R1	x	x	x	x	-	-	-	-
….	x	x	x	x	x	-		
Rn	x	x	-				x	
Totales								

Tabla 7. Estados posibles de los requisitos.

En la Tabla 7, en lugar de registrar una "x", para conocer el estado del requisito es recomendable registrar el documento o producto que se genera [5].

Debemos incorporar dentro de la línea base el número total de cambios propuestos y aprobados [6]. La Figura 14 nos muestra algunos de los estados incluidos en las fases del proyecto. Es decir, se estableció en la linea base de la Fase 1 con 20 requisitos, de los cuales en el avance del proyecto se han añadido 10, modificados 5, y se han anulado 5 de los requisitos.

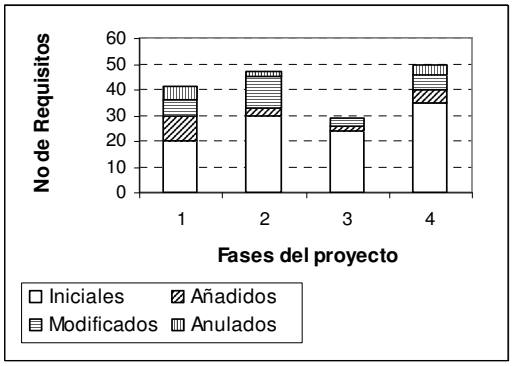


Figura 14. Cantidad total del estado de requisitos en las fases del proyecto

4. El Cliente Frente a los Requisitos

Los clientes necesitan cambios y el gestor de proyectos debe responder a estos cambios. A medida que el proyecto informático va desarrollándose el cliente comienza a adquirir mayor conocimiento y experiencia, e intentará modificar los requisitos iniciales, significando que los cambios pueden suscitarse en cualquier momento del ciclo de vida del proyecto.

4.1 Participación del cliente

Los clientes ocupados podrían preferir no llegar a involucrarse en la ingeniería de requisitos. Sin embargo la falta de no involucrarlos, incrementa grandemente el riesgo de construir un producto equivocado [12]. El éxito de participación del cliente puede aumentar el éxito de la implementación y posible aceptación del sistema final. Debemos buscar que su participación apunte a contribuciones que hubieran sido erradas sino participaba. Un proyecto normal experimenta alrededor del 25 por ciento de cambios en los requisitos a lo largo de su desarrollo, lo que añade más de un 25 por ciento de esfuerzo al proyecto [2]. El cliente es un factor crítico en la entrega excelente del software. Si entregamos el producto sin realizar antes alguna reunión con el cliente puede darse la posibilidad de que el cliente descubra que el sistema solicitado no reúna su necesidades, implicando realizar nuevos cambios o modificaciones (Figura 15), con gran impacto económico, iniciando el replanteo desde las primeras fases del proyecto. En tal sentido, se debe establecer un conjunto de reuniones con el cliente con la finalidad de verificar si se están cumpliendo con los requisitos. Estas reuniones antes de la entrega del producto podrán verificar la satisfacción del cliente, si se realiza algún cambio. El gestor del proyecto ha de confeccionar resúmenes semanales o mensuales, según el criterio de establecer dichas reuniones, señalando lo que el equipo ha realizado y lo que plantea realizar la siguiente semana. Estos reuniones establecidas con el cliente (Tabla 8) ayudarán a verificar si el equipo está desarrollando los respectivos requisitos y tomar las medidas correctivas sino es el caso [1]. Una de las formas más efectivas de mostrar que se han recibido y comprendido sus requisitos es mostrarles en la siguiente reunión los cambios efectuados. Asimismo, el cliente debe conocer las consecuencias de su participación por el incremento o modificación de los requisitos que suscitan el incremento de costos,tiempo de duración y personal (Sección 3.3.3).

Personal del Proyecto	Período de Reunión			
	Fase1	Fase2	Fase3	Fase4
Director	X			
Jefe de Proyecto	X	X		
Analista		X	X	X
Diseñador			X	X
Codificador				X
.....				
Cliente	X	X	X	X
Resumen	Doc1	Doc2	Doc3	Doc4

Tabla 8. Reuniones con el cliente para verificar cumplimiento de los requisitos

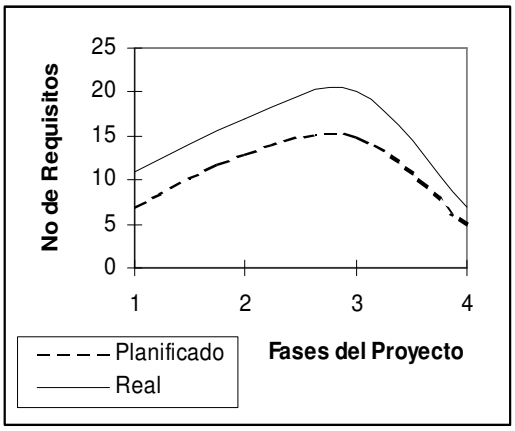


Figura 15. Incremento de requisitos por fase

4.2 Satisfacción

Las empresas de hoy en día no pueden dejar de conocer la satisfacción del cliente, es necesario saber si estamos cumpliendo con sus exigencias y expectativas. Muchas de las grandes compañías de éxito han identificado la importancia de enfocar la satisfacción de los clientes [11]. Para conocer dicha satisfacción se selecciona un conjunto de atributos (Figura 16), dentro de los cuales hemos seleccionado el cumplimiento de los requisitos. Para establecer estos resultados existe un conjunto de técnicas o herramientas que seleccionará cada compañía en función de sus prioridades, entre las técnicas se encuentra la realización de encuestas o entrevistas no estructuras.

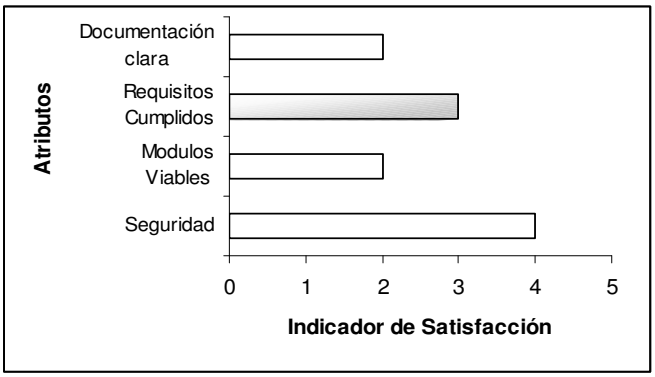


Figura 16. Atributos de satisfacción

5. Conclusiones

En este artículo se presenta un conjunto de gráficas y tablas con la finalidad de que el gestor de proyectos y los stakeholder implicados en la gestión de requisitos puedan tener una visión más objetiva de las repercusiones que se originan cuando un requisito es cambiado.

Consideramos necesario representar la gestión de requisitos mediante gráficas y tablas con la finalidad de obtener una visión e información rápida de los cambios de requisitos que se desarrollan en el transcurso del proyecto, para conocer el origen de los cambios, motivo del mismo, sus versiones, el impacto con respecto a otros requisitos vinculados y como todos estos cambios influyen en los costes, plazos y recursos.

Aunque el requisito no sólo es modificado por el equipo del proyecto, es modificado también por el cliente, en tal sentido, es importante que el cliente conozca las variaciones que se vienen realizando en el proyecto cada vez que cambia un requisito.

Se pretende abarcar el verdadero estado del cambio de requisitos, para controlar la situación real en las diferentes fases del ciclo de vida del proyecto.

Dentro de las normas y modelos de estandarización ISO/IEC 15504, el CMM y el PMBOK existen un conjunto de procesos textuales dirigidos a la gestión de requisitos, por lo que hemos considerado conveniente presentar dichos procesos en forma visual cumpliendo los objetivos mencionados en dichos estándares.

Referencias

[1] Cuevas G.A. *Gestión del Proceso Software.* Centro de Estudios Ramón Areces. Madrid (España), 2002

[2] Jones, C.: *Assesment and Control of Software Risks*, 1994.

[3] IEEE STD-610. Computer Dictionary. *Compilation de IEEE Standard Computer Glossaries.* IEEE Computer Society, 1990

[4] ISO/IEC 15504. *Information technology – Software process assessment.* 1998.

[5] Kaputo Kim. *CMM Implementation Guide: Choreographing Software Process Improvement,* 1998.

[6] Leite Sampaio do Prado y Doom Jorge Horacio. *Perspectives on software requirements*. Kluwer Academic Publishers, 2004.

[7] Letelier Patricio. *A Framework for Requirements Traceability in UML-based Projects. 1st International Workshop on Traceability in Emerging Forms of Software Engineering. In conjunction with the 17th IEEE International Conference on Automated Software Engineering*. Septiembre 2002

[8] Paulk, M., Weber, Ch., Garcia, S., Chrissis, M. and Bush, M.*Capability Maturity Model for Software*, Versión 1.1, Technical Report CMU/SEI 93 TR-25, February 1993.

[9] PMI, Project Management Institute. *A Guide to the Project Management Body of Knowledge*. PMBOK, 2000.

[10] Sommerville, I. and Sawyer P. *Requerimentes Engineering: A Good Practice Guide*. Johm Wiley and Sons, 1997

[11] Vavra T.G. *Improving Your Measurement of Customer Satisfaction: A Guide to Creating, Conducting, Analyzing, and Reporting Customer Satisfaction Measurement Programs.* ASQC Quality Press. Milwaukee,1997.

[12] Wiegers Karl E. *Software Requirements: Practical techniques for gathering and managing requirements throughout the product development cycle.* Microsoft Press, 2003.

Uma Ferramenta de Apoio ao Controle de Versão das Aplicações Criadas por um Framework

Maria Istela Cagnin*, José Carlos Maldonado, Rosana T. V. Braga, Fernão Germano
Universidade de São Paulo,
Instituto de Ciências Matemáticas e de Computacão,
São Carlos, São Paulo, Brasil,
Caixa Postal 668, CEP 13560-970,
{istela, jcmaldon, rtvb, fernao}@icmc.usp.br

Rosângela Penteado
Universidade Federal de São Carlos,
Departamento de Computacão,
São Carlos, São Paulo, Brasil,
Caixa Postal 676, CEP 13.565-905,
{rosangel}@dc.ufscar.br

Abstract

Framework based application development is increasingly being adopted by software organizations. Frameworks provide reuse of both software design and code, and supply more trustable applications, as the components used to implement them have been previously tested. However, version control is more problematic than in conventional software development, as it is necessary to control both the framework versions and the versions of the applications created with it. Furthermore, aiming to minimize the impact of system requirement changes, framework based software development and reengineering processes adopt the incremental approach, which is a "must" in agile methodologies. This approach makes easier to fulfill requests for system requirements change at any time during the process application. In that context, there is a lack of tools that support version control of applications created with frameworks. This paper presents a tool that aims to aid in the fulfillment of that need, contributing to quality assurance of the products that result from software development or reengineering.

Keywords: Frameworks, Incremental Reengineering, Incremental Development, Version Control Tool.

Resumo

O desenvolvimento de aplicações baseado em frameworks vem sendo praticado cada vez mais pelas empresas de software. Frameworks proporcionam reuso de projeto e de código, e fornecem aplicações mais confiáveis, uma vez que seus componentes, utilizados para construí-las, foram testados anteriormente. No entanto, a problemática do controle de versão é mais grave do que no desenvolvimento de software convencional, pois é necessário controlar tanto as versões do framework quanto as das aplicações por ele criadas. Além disso, visando minimizar o impacto das mudanças que ocorrem nos requisitos do sistema, processos de desenvolvimento e de reengenharia de software baseados em frameworks adotam a abordagem incremental, que é indispensável a metodologias ágeis. Essa abordagem facilita o atendimento das solicitações de mudanças nos requisitos do sistema, em qualquer momento da aplicação do processo. Nesse contexto, há carência de ferramentas que apóiem o controle de versão das aplicações criadas por frameworks. Este artigo apresenta uma ferramenta que tem como objetivo amenizar essa carência colaborando para a garantia da qualidade do produto resultante do desenvolvimento ou da reengenharia de software.

Palavras-Chaves: Frameworks, Reengenharia Incremental, Desenvolvimento Incremental, Ferramenta de Controle de Versão.

*Apoio Financeiro da FAPESP - nr. 00/10881–4

1 Introdução

O software evolui constantemente para se adequar às mudanças administrativas, funcionais, do negócio, governamentais, dentre outras, visando a satisfazer às necessidades dos usuários. Para controlar tais mudanças, é necessário estabelecer um controle das versões produzidas e entregues aos usuários de maneira sistemática e efetiva. Para isso, é necessário considerar durante todo o processo de desenvolvimento/reengenharia de software o *Software Configuration Management - SCM*, que é um conjunto de atividades que gerencia mudanças durante todo o ciclo de vida do software e é um elemento importante para garantir a sua qualidade [23].

O desenvolvimento de aplicações baseado em frameworks [27, 12] vem sendo praticado cada vez mais pelas empresas de software. Um framework é um conjunto de classes que incorpora um projeto abstrato e reusável de soluções para uma família de problemas relacionados em um domínio particular [18], proporcionando reuso de projeto e de código, e fornecendo aplicações mais confiáveis. Isso porque seus componentes, utilizados para construir as aplicações, foram testados anteriormente. Além disso, Fayad e Schmidt afirmam que aplicações podem ser desenvolvidas muito mais rapidamente e com mínimo esforço [12].

No entanto, a problemática de controle de versão no contexto de desenvolvimento baseado em framework é mais grave do que no desenvolvimento de software convencional, pois é necessário controlar as versões do framework e também as aplicações geradas a partir dele.

O controle de versão de frameworks deve ser realizado com muita precaução, pois evoluções em frameworks podem mudar o seu projeto e, conseqüentemente, o das suas aplicações dependentes. Como resultado, essas aplicações podem não aderir ao novo projeto e podem deixar de fornecer o comportamento desejado.

Visando minimizar o impacto das mudanças que ocorrem nos requisitos do sistema, alguns processos de desenvolvimento e de reengenharia de software, baseados em frameworks, adotam a abordagem incremental, que é indispensável a metodologias ágeis [1, 2, 30]. Essa abordagem facilita o atendimento das solicitações de mudanças nos requisitos do sistema, em qualquer momento da aplicação do processo.

Nesse contexto, há carência de ferramentas que apóiem tanto o controle de versão de frameworks quanto o controle de versão de aplicações criadas por eles. A instanciação de frameworks permite que novos componentes sejam adicionados e/ou retirados de uma aplicação criada previamente. No entanto, caso a aplicação tenha sido modificada com a adição manual de novas funcionalidades, não fornecidas pelo framework, e houver necessidade de instanciar o framework posteriormente para adicionar outros componentes na aplicação, todas as linhas de código inseridas manualmente serão perdidas.

Este artigo apresenta uma ferramenta, específica para o controle de versões das aplicações de um determinado framework, que tem como objetivo amenizar essa carência e, também, apoiar na realização da tarefa *controle de versão* da atividade SCM, colaborando para garantir a qualidade do produto resultante tanto no desenvolvimento quanto na reengenharia.

Na Seção 2, discutem-se os trabalhos relacionados. Na Seção 3 apresenta-se a ferramenta, denominada *GREN-WizardVersionControl*, quanto aos aspectos de modelagem conceitual, de arquitetura e de implementação. Na Seção 4 relata-se experiência de uso da ferramenta em um estudo de caso de reengenharia. Na Seção 5, discutem-se as conclusões e os trabalhos futuros.

2 Trabalhos Relacionados

O SCM é considerado como uma das atividades importantes para atingir a certificação de qualidade em diversos padrões, como ISO 9000 [16], *Capability Maturity Model* (CMM) [22], *Capability Maturity Model Integration* (CMMI) [8] e ISO/IEC 15504 (*Software Process Improvement and Capability dEtermination*) ou SPICE [17]. De acordo com Pressman [23], SCM é composto por cinco tarefas: (1) identificação das mudanças, (2) controle de versão, (3) controle de mudanças, (4) auditoria de configuração, e (5) relatórios (que apresentam o que aconteceu, quem fez a mudança, quando aconteceu a mudança, o que será afetado com a mudança, etc). A tarefa *controle de versão* é a mais conhecida e a que possui a principal responsabilidade, pois trata do armazenamento e da recuperação de diferentes versões dos artefatos gerados durante o processo de software.

Algumas características desejáveis de um sistema de SCM são ressaltadas por Midha [20]: a) facilidade de uso, para que o usuário possa utilizar as funcionalidades da ferramenta como parte da execução de seu trabalho; b) facilidade de administração da ferramenta; c) suporte ao desenvolvimento distribuído, para que o sistema possa ser utilizado também por equipes de desenvolvimento distribuídas remotamente; e d) integração da funcionalidade de SCM com outras ferramentas.

A fim de minimizar o espaço de armazenamento das versões dos artefatos no repositório, os sistemas de SCM utilizam **delta** *scripts*, ou somente **delta**, no qual somente uma versão do artefato é armazenada integralmente, e as demais versões possuem apenas as diferenças armazenadas [25]. Existem duas abordagens para o tratamento do delta: **delta negativo** e **delta positivo**. A primeira abordagem, também chamada de **delta reverso**, armazena integralmente a versão mais recente e as diferenças até então, disponibilizando de forma mais rápida a última versão.

A abordagem **delta positivo** armazena integralmente a versão mais antiga e as diferenças desde então. A ferramenta apresentada neste artigo armazena apenas as modificações realizadas em cada versão das aplicações. Isso porque existe um histórico de instanciação do framework, que está sendo considerado pela ferramenta apresentada neste artigo, que armazena as funcionalidades que compõem cada aplicação gerada. Com isso, é possível obter automaticamente informação da versão mais antiga da aplicação.

Existem diversas ferramentas de SCM, com várias funcionalidades, diferentes complexidades e preços. Portanto, é necessário que o engenheiro de software avalie cada uma e escolha a mais apropriada para as suas necessidades. *ClearCase* [15], *Continuus/CM* [28], *Visual Source Safe* [19], *QVCS* [24], *FreeVCS* [14], *CVS* [11], *VersionWeb* e *PVCS* [26] são algumas das ferramentas de SCM existentes. Nenhuma delas atende a necessidade de controlar versões de aplicações geradas por frameworks. No entanto, foi encontrada uma ferramenta, proposta por Tourwé [29], com objetivo diferente da ferramenta apresentada neste artigo, que controla as mudanças causadas na evolução de frameworks e das aplicações criadas a partir deles. Tal ferramenta avalia o impacto que as mudanças podem ter, tanto na hierarquia de classes do framework quanto na das aplicações dependentes dele. Para isso: a) fornece previamente a definição da propagação de mudanças, permitindo que o desenvolvedor avalie o impacto das transformações no framework e de suas aplicações dependentes e detecte possíveis conflitos de "*merge*"; b) sugere como esses problemas podem ser resolvidos.

3 Ferramenta *GREN-WizardVersionControl*

A necessidade de criar a ferramenta *GREN-WizardVersionControl* foi evidenciada em um estudo de caso de reengenharia de um sistema legado de controle de biblioteca [6, 9], e em outro referente ao controle de oficina eletrônica [7], ambos desenvolvidos em Clipper. A reengenharia desses sistemas foi realizada com o apoio do processo de reengenharia ágil PARFAIT [1] [7].

PARFAIT é incremental e iterativo, pois o engenheiro de software tem a flexibilidade de retornar a passos do processo anteriormente executados, a fim de refinar os artefatos produzidos. Além disso, é considerado um processo ágil, pois atende a diversas *práticas* de metodologias ágeis [2]: a) fornece aos usuários uma versão do novo sistema o mais rápido possível – atende à *prática "pequenas releases" do XP (eXtreme Programming)*; b) os usuários participam ativamente da maioria das atividades do projeto de reengenharia e aprovam o produto à medida que o projeto evolui. Em cada interação com os usuários, melhorias no produto são realizadas – atende à *prática "clientes presentes" do XP*; c) testes no novo sistema são realizados constantemente e comparados com os resultados dos testes do sistema legado – atende à *prática "testes constantes" do XP*; d) o planejamento da reengenharia é refeito a cada iteração do processo – atende à *prática "jogo do planejamento" do XP*; e e) incentiva a *prática "programação em pares" do XP*.

PARFAIT utiliza o framework GREN ("Gestão de REcursos de Negócios") [5] como apoio computacional: a) para as atividades de engenharia reversa: na elicitação de novos requisitos e na identificação de regras de negócio do sistema legado; e b) para as atividades de engenharia avante, na criação do protótipo do novo sistema. A construção desse framework foi baseada na linguagem de padrões de análise GRN (Gestão de Recursos de Negócios) [3], portanto facilita a reengenharia de sistemas legados no domínio de "Gestão de Recursos de Negócios". O framework foi construído utilizando a linguagem de programação Smalltalk e o SGBD MySQL [21].

Para facilitar a instanciação do GREN, há um ferramenta de apoio, denominada GREN-Wizard [4], que solicita os padrões da GRN usados na modelagem do sistema e gera as classes da nova aplicação em Smalltalk e as tabelas no banco de dados relacional MySQL.

Após a criação da aplicação de controle de biblioteca, durante a reengenharia com o apoio do GREN-Wizard, algumas alterações foram realizadas em seu código fonte a fim de torná-la funcionalmente mais semelhante ao legado.

Além disso, o cliente solicitou a adição de uma funcionalidade que não estava presente no sistema legado, que foi a de *"Cobrança de Taxa de Multa"* na devolução do livro, caso o usuário devolva o livro com atraso. Tal funcionalidade é suportada por um dos padrões da GRN e, portanto, é fornecida pelo framework GREN. Logo, não precisaria ser implementada manualmente. Com isso, bastaria apenas criar uma nova versão da aplicação no GREN, adicionando-lhe a funcionalidade solicitada. Mas, isso não era possível pois, com uma nova instanciação do framework, todas as linhas de código fonte, inseridas anteriormente na aplicação, seriam perdidas. Dessa forma, evidenciou-se a necessidade de criar uma ferramenta que pudesse armazenar, em uma meta-base de dados, todas as alterações realizadas no código fonte das aplicações geradas pelo framework para que, em uma próxima instanciação do framework, todas as modificações realizadas no código fonte, em versões anteriores, pudessem ser incorporadas na nova versão da aplicação.

Além de evidenciar a necessidade de criar a ferramenta de controle de versão das aplicações criadas pelo framework GREN, observou-se também a necessidade de evoluir a ferramenta de instanciação GREN-Wizard, para que ela pudesse incorporar, durante a criação da nova versão da aplicação, as alterações realizadas no código fonte das versões anteriores. Isso possibilita que o framework GREN apóie efetivamente os processos de desenvolvimento e os

[1] Processo Ágil de Reengenharia baseado em FrAmework no domínio de sistemas de Informação com VV&T

de reengenharia incremental, tornando possível adicionar novas funcionalidades, em qualquer momento da aplicação desses processos.

Assim, a ferramenta foi criada visando que a agilidade do processo de reengenharia PARFAIT não seja comprometida, já que usa a abordagem incremental na fase de implementação. Essa ferramenta é útil não apenas na aplicação do PARFAIT como também no desenvolvimento de novas aplicações de modo incremental.

3.1 Modelagem Conceitual

Esta seção tem como objetivo apresentar as funcionalidades da ferramenta *GREN-WizardVersionControl* e o seu meta-modelo, por meio de um diagrama de casos de uso e de um diagrama de classes, respectivamente. Ambos diagramas são provenientes da UML (*Unified Modelling Language*) [13].

3.1.1 Funcionalidades da Ferramenta

O diagrama de casos de uso da ferramenta *GREN-WizardVersionControl* é mostrado na Figura 1. Todos os casos de uso, com exceção do caso de uso com preenchimento cinza ("Decidir conflito entre métodos"), são executados durante as modificações no código fonte da aplicação gerada pelo framework. O caso de uso "Decidir conflito entre métodos" é executado durante a instanciação do framework com o apoio da ferramenta GREN-Wizard, quando algum método herdado do framework foi modificado pelo engenheiro de aplicação [2]. O caso de uso "Alterar atributo" é somente executado para atributos primitivos (`inteiro, string, float, date`) e não herdados do framework. Além de adicionar atributos primitivos (caso de uso "Adicionar Atributos") é possível adicionar atributos do tipo enumerado (caso de uso "Adicionar atributo enumerado") e multivalorado (caso de uso "Adicionar atributo multivalorado"), que possuem tratamentos especiais. Os casos de uso "Remover classe", "Remover atributo", "Remover categoria método" e "Remover método" são executados somente para classes, atributos, categoria de métodos e métodos não herdados do framework, respectivamente.

As restrições de remoção foram estabelecidas a fim de garantir que os componentes herdados do framework (ou seja, classes, atributos, métodos, etc) não sejam removidos acidentalmente ou propositadamente do código fonte da aplicação criada pelo framework. Só é permitido que o engenheiro de aplicação sobreponha ou modifique os métodos herdados, garantindo assim que nenhum *hot-spot* [3] seja danificado, garantindo que a estrutura da aplicação esteja correta e funcionando adequadamente.

3.1.2 Meta-Modelo da Ferramenta

O meta-modelo da ferramenta *GREN-WizardVersionControl* é apresentado por meio de um diagrama de classes, como ilustrado na Figura 2.

Algumas classes existentes do framework GREN (`PersistentObject`) e da ferramenta de instanciação GREN-Wizard (`GrenWizardCodeGenerator`, `GrenWizardGUI`, `GrenApplication`, `AttributeMappingForm` e `AttributeListForm`) tiveram que ser modificadas, conforme ilustrado no diagrama as classes com preenchimento. As classes `ClassVersionControl`, `ClassProtocolVersionControl` e `ClassMethodVersionControl` contém informações relevantes sobre a classe que está sendo atualizada, sobre o protocolo ou categoria [4] do método e sobre o método, respectivamente. Como essas classes contém algumas informações em comum, elas herdam da classe `MetaVersionControl`. A classe `GREN-WizardVersionControl` faz todo o controle de versão e contém uma interface semelhante ao *System Browser*, que é uma das ferramentas do VisualWorks [10] utilizada para realizar a programação em Smalltalk. A classe `GrenWizardDifferatorTool` [5] mostra os trechos de código diferentes dos métodos herdados do framework mas que foram modificados. Isso é feito durante a instanciação de uma nova versão da aplicação por meio da ferramenta GREN-Wizard.

3.2 Arquitetura

A arquitetura da ferramenta *GREN-WizardVersionControl* foi baseada na arquitetura do framework GREN e da ferramenta de instanciação GREN-Wizard.

A arquitetura do GREN foi projetada em três camadas: a de persistência, a de negócios e a de interface gráfica com o usuário (GUI), conforme apresentado na Figura 3. A **camada de persistência** possui classes que tratam da conexão com a base de dados, gerenciamento dos identificadores dos objetos e persistência dos objetos. A **camada**

[2] suas funções estão apresentadas na Seção 3.2

[3] estruturas variáveis do framework que contém os componentes que podem ser adaptados e estendidos pelo engenheiro de software na aplicação instanciada a partir do framework, para se adequarem às necessidades de um determinado sistema (por exemplo, regras do negócio, requisitos funcionais inerentes às políticas internas da empresa, etc).

[4] em Smalltalk, métodos da classe são organizados em protocolos (ou categorias) que são agrupamentos de métodos semanticamente relacionados.

[5] baseada na classe `Differator` do VisualWorks

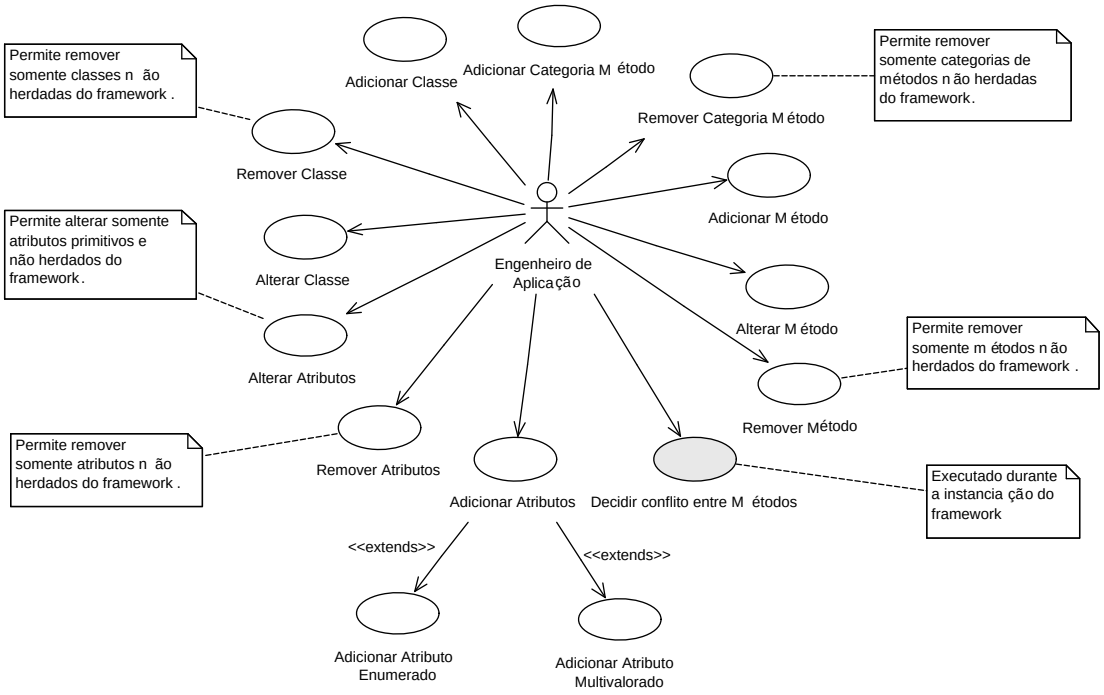


Figura 1: Diagrama de Casos de Uso da ferramenta *GREN-WizardVersionControl*

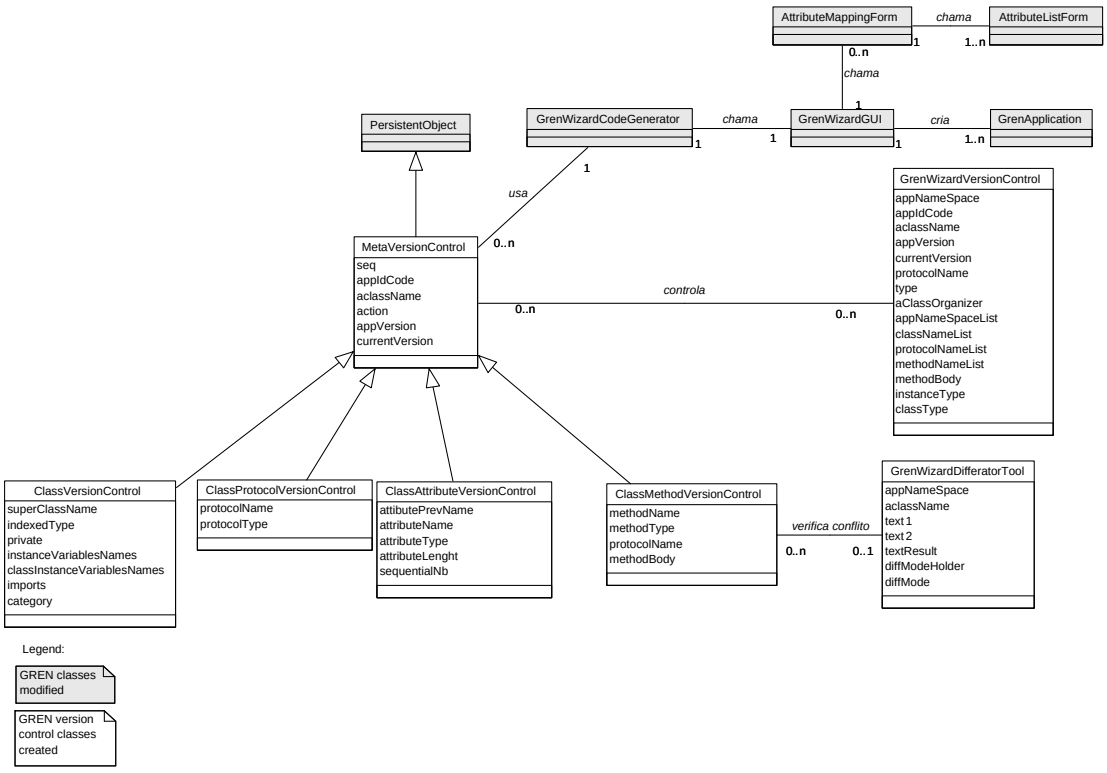


Figura 2: Diagrama de Classes da ferramenta *GREN-WizardVersionControl*

de negócios comunica-se com a camada de persistência para armazenar um objeto. Na camada de negócios existem diversas classes derivadas diretamente dos padrões de análise que compõem a linguagem de padrões GRN, ou seja, as classes e relacionamentos contidos em cada padrão possuem a implementação correspondente nesta camada. A **camada de interface gráfica com o usuário** contém as classes que tratam a entrada e a saída de dados, permitindo a interação do usuário final com o sistema. Essa camada comunica-se com a camada de negócios para obtenção de objetos a serem mostrados na interface com o usuário e para devolver informações a serem processadas pelos métodos da camada de negócios.

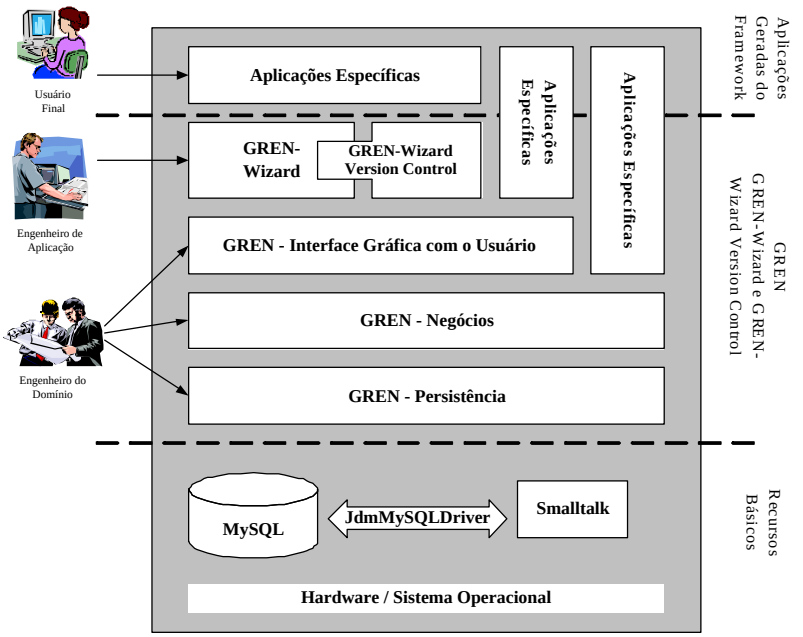


Figura 3: Arquitetura do framework GREN (adaptada de [5])

A camada GREN-Wizard, que gera o código da aplicação a partir da sua especificação baseada nos padrões da GRN, situa-se acima da camada do GREN, pois utiliza todas as demais camadas. A camada GREN-Wizard comunica-se com a camada *GREN-WizardVersionControl*, que é responsável pelo controle das versões das aplicações instanciadas e apóia o desenvolvimento e reengenharia incremental e, conseqüentemente, processos baseados em metodologias ágeis. Aplicações específicas podem ser construídas a partir da camada de interface gráfica com o usuário, sem o apoio da camada GREN-Wizard, usando (por meio de herança ou referência a objetos) classes de todas as camadas do GREN inferiores a essa, ou podem ser construídas com base na camada de negócios, caso a camada de interface gráfica com o usuário não seja reutilizada a partir do GREN, mas implementada separadamente.

A Figura 4 mostra a arquitetura das ferramentas GREN-Wizard e *GREN-WizardVersionControl*. Dois atores interagem diretamente com a ferramenta GREN-Wizard: o engenheiro do domínio, responsável pela construção do GREN-Wizard (e, provavelmente, também pelo desenvolvimento da linguagem de padrões e por parte da construção do framework) e o engenheiro de aplicações, que utiliza o GREN-Wizard para gerar aplicações específicas por meio de uma interface gráfica com o usuário (GUI). Essa interface facilita a especificação da aplicação de acordo com os padrões da linguagem de padrões GRN. O usuário final interage indiretamente com a ferramenta GREN-Wizard, pois executa o código da aplicação específica, gerado por ela.

Três módulos compõem a arquitetura do GREN-Wizard: o módulo de especificação do domínio, o módulo de especificação da aplicação e o módulo de geração de código.

O módulo de **especificação do domínio** permite a representação de informações sobre a linguagem de padrões GRN e seu mapeamento para as classes do framework GREN. O módulo de **especificação da aplicação** é responsável por armazenar as informações sobre a modelagem das aplicações específicas usando a linguagem de padrões. Por fim, o módulo de **geração de código** baseia-se nas informações disponibilizadas sobre a especificação do domínio e sobre a especificação da aplicação, para gerar o código específico da aplicação, que deve juntar-se ao código do framework para produzir a aplicação a ser entregue ao usuário final.

De acordo com a Figura 4, o engenheiro de aplicação deve utilizar inicialmente a ferramenta GREN-Wizard para gerar uma primeira versão da aplicação. Nessa ferramenta, como mencionado anteriormente, ele especifica a aplicação informando os padrões da GRN utilizados para modelá-la. Qualquer modificação manual no código fonte da aplicação, a fim de inserir novas funcionalidades não fornecidas pelo framework, deve ser realizada por meio da ferramenta *GREN-WizardVersionControl*, que armazena cada modificação em uma meta-base de dados. Quando houver

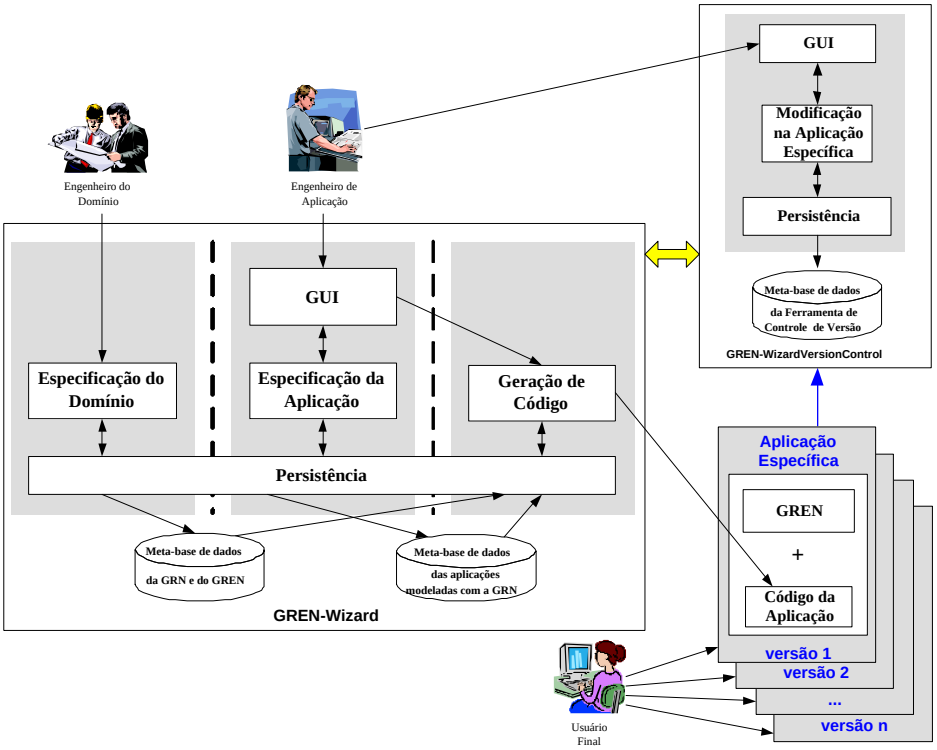


Figura 4: Arquitetura das ferramentas GREN-Wizard e GREN-WizardVersionControl

necessidade de inserir novas funcionalidades na aplicação, fornecidas pelo framework, o engenheiro de aplicação deve utilizar a ferramenta GREN-Wizard. Essa ferramenta gera a nova versão da aplicação, com as novas funcionalidades herdadas, juntamente com todas as modificações realizadas anteriormente no código fonte.

O usuário final interage com todas as versões da aplicação liberadas para uso, ou seja, as *releases*.

3.3 Implementação

A ferramenta *GREN-WizardVersionControl* foi implementada na mesma linguagem de programação em que o framework GREN e a ferramenta GREN-Wizard estão implementados, ou seja, Smalltalk (ambiente VisualWorks), para que essas ferramentas pudessem ser integradas da melhor maneira possível. Isso atende uma das características desejáveis de um sistema de SCM ressaltadas por Midha [20]. O meta-modelo foi mapeado para uma meta-base de dados no SGBD MySQL [21].

A Figura 5 apresenta a tela principal da ferramenta *GREN-WizardVersionControl*, cujo projeto de interface foi baseado na tela do *System Browser* do VisualWorks, a fim de torná-la o mais similar possível, objetivando facilitar o uso da ferramenta e possibilitar que o usuário utilize as funcionalidades da ferramenta como parte da execução de seu trabalho. Isso atende outra característica desejável de um sistema de SCM [20].

A administração da ferramenta *GREN-WizardVersionControl* é fácil, colaborando para que mais uma característica desejável de um sistema de SCM [20] seja alcançada.

Como mencionado anteriormente, além da criação da ferramenta *GREN-WizardVersionControl* foi necessário também alterar a ferramenta GREN-Wizard, para que as alterações realizadas pelo engenheiro de software no código fonte de uma determinada aplicação fossem consideradas também nas próximas instanciações do framework para tal aplicação. Na geração de uma nova versão da aplicação pelo GREN-Wizard, as classes, métodos e atributos que foram removidos, inseridos ou atualizados na versão anterior da aplicação com o apoio da ferramenta *GREN-WizardVersionControl* e que estão armazenados em sua meta-base de dados, são detectados. As inserções e remoções de código fonte são resolvidas automaticamente pela ferramenta GREN-Wizard. No entanto, as modificações nos métodos herdados do framework são identificadas pela ferramenta como conflito entre o código fonte do método do framework e o da aplicação e deve ser resolvido pelo engenheiro de software em tempo de execução do GREN-Wizard. Para isso, quando a ferramenta GREN-Wizard detecta conflitos, é apresentada uma tela para o engenheiro da aplicação, Figura 6 (lado esquerdo), que deve informar o conteúdo do método que a ferramenta de instanciação deve considerar na versão da aplicação sendo criada. Para isso, é necessário copiar e colar o conteúdo correto na caixa de texto *Resulting Method*, Figura 6 (lado direito). Em seguida, deve-se clicar no botão *Accept* para compilar as linhas de código fonte informadas e para dar prosseguimento a instanciação do framework.

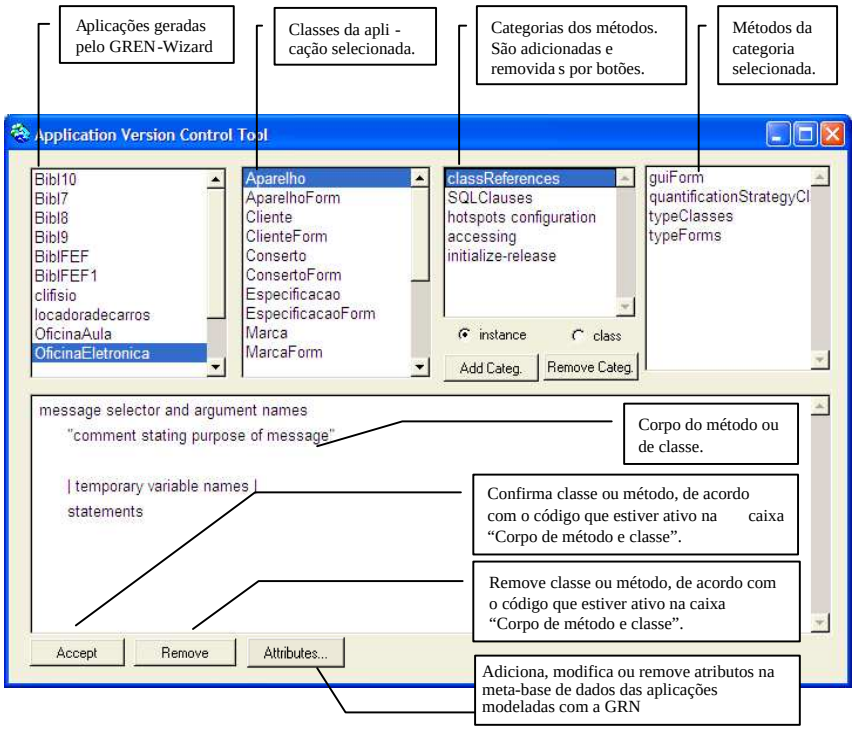


Figura 5: Tela principal da ferramenta *GREN-WizardVersionControl*

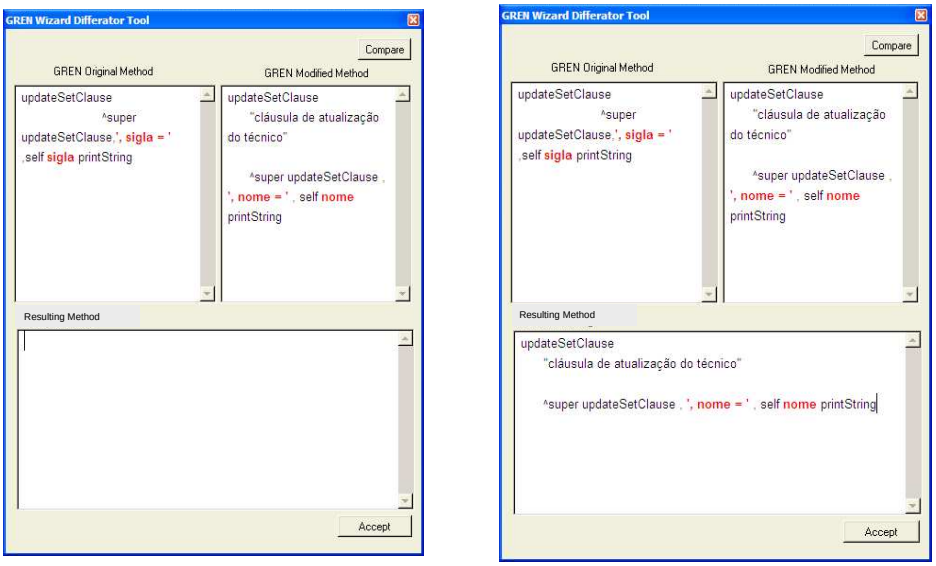


Figura 6: Tela para solução de conflito entre métodos do framework e da aplicação

A ferramenta GREN-Wizard também cria a base de dados e os *scripts* das tabelas em MySQL da aplicação que foi gerada. Uma alteração nessa funcionalidade também foi realizada, permitindo que o engenheiro de aplicação tenha a opção de gerar uma nova base de dados da aplicação ou manter a atual e apenas atualizar a estrutura modificada, a fim de não perder os dados da aplicação inseridos na versão anterior.

A ferramenta *GREN-WizardVersionControl* foi testada em outro estudo de caso de reengenharia do sistema de controle de biblioteca, com o apoio do PARFAIT, e mostrou-se efetiva, como apresentado na próxima seção.

4 Exemplo de Uso da Ferramenta *GREN-WizardVersionControl*

A Figura 7 apresenta a tela "Empréstimo" de livro e o script SQL da tabela `Emprestimo` da primeira versão do sistema de controle de biblioteca, ambos gerados pelo GREN-Wizard durante a reengenharia com o apoio do processo

ágil PARFAIT.

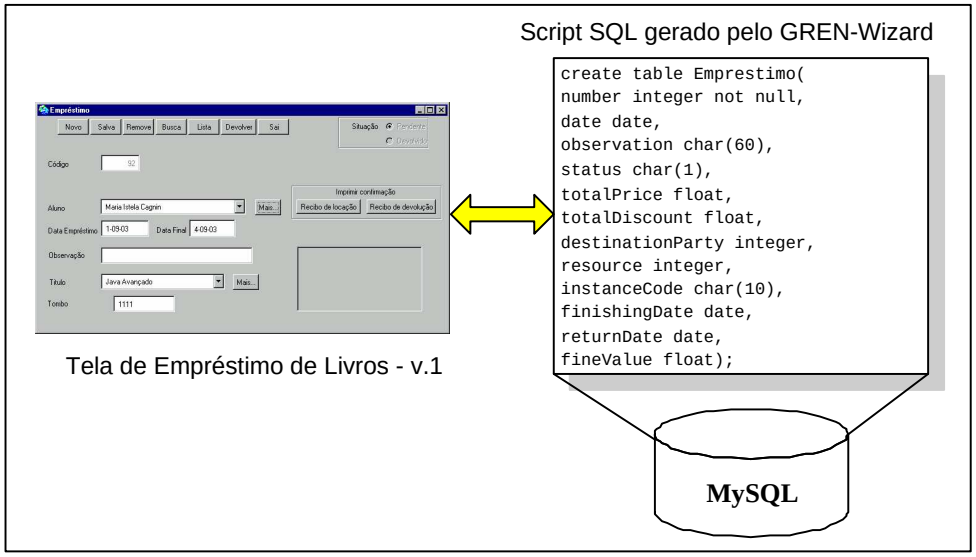


Figura 7: Fragmento da Versão 1 da aplicação de controle de biblioteca

No decorrer da aplicação do processo de reengenharia, observou-se a necessidade de realizar algumas mudanças no código fonte da primeira versão da aplicação. Para isso, utilizou-se a ferramenta *GREN-WizardVersionControl*. A Figura 8 apresenta uma das mudanças realizadas, a qual modificou o conteúdo do método `windowLabel`, herdado do framework GREN: do conteúdo original `^'Emprestimo'` para o conteúdo modificado `^'Empréstimo de Livro'`. Essa modificação foi armazenada na meta-base de dados da ferramenta *GREN-WizardVersionControl*, Figura 8, para que pudesse ser incorporada pelo GREN-Wizard na criação da próxima versão da aplicação.

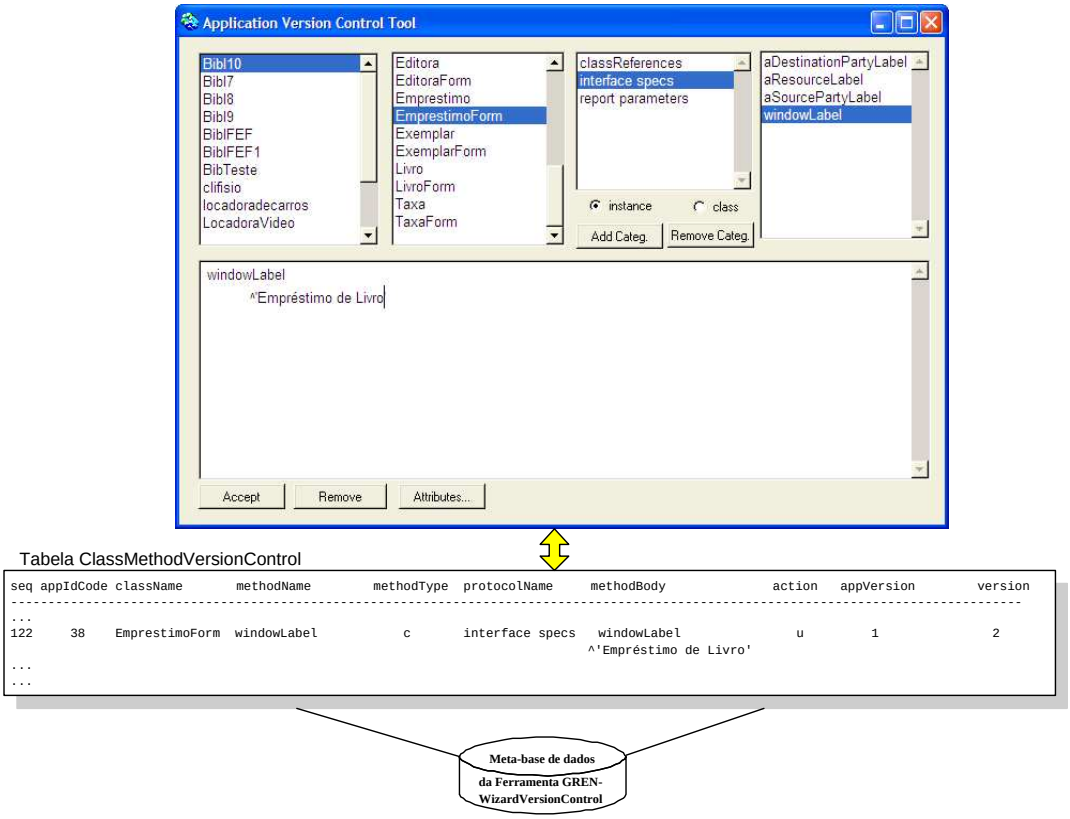


Figura 8: Mudança no código fonte da versão 1 da aplicação de controle de biblioteca

Outra mudança desejada na aplicação foi a adição da funcionalidade *"Cobrança de Taxa de Multa"*. Nesse

caso, utilizou-se a ferramenta GREN-Wizard, uma vez que essa funcionalidade é fornecida pelo GREN. Durante a instanciação da nova versão da aplicação, a ferramenta GREN-Wizard detectou a modificação realizada no método `windowLabel` como conflito entre o conteúdo do método herdado do framework e o da aplicação (Figura 9). Com isso, o engenheiro de aplicação teve que solucionar esse conflito, por meio da tela *GREN-Wizard Differator Tool* (Figura 9), apresentada pelo GREN-Wizard em tempo de execução da instanciação.

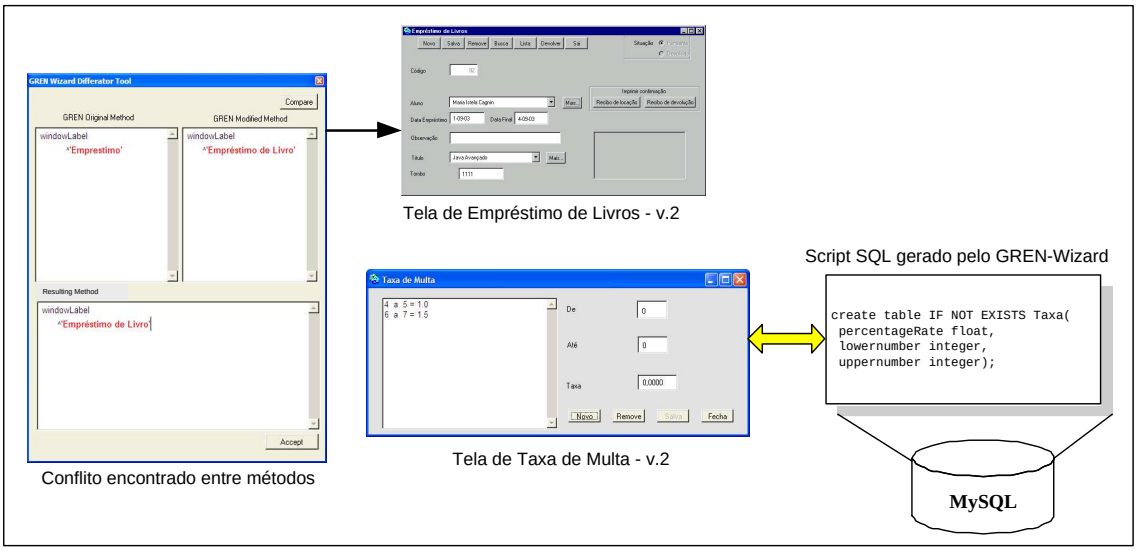


Figura 9: Fragmento da Versão 2 da aplicação de controle de biblioteca

A Figura 9 também apresenta algumas telas da nova versão do sistema de controle de biblioteca: a tela "Empréstimo de Livros", tendo o título da janela atualizado, e a tela "Taxa de Multa", resultante da funcionalidade adicionada por meio do GREN-Wizard, bem como o seu respectivo script SQL.

5 Conclusões

Observou-se que a ferramenta *GREN-WizardVersionControl* apóia processos de reengenharia e de desenvolvimento de sistemas baseados em framework, que utilizam abordagem incremental, pois, por meio dela, o engenheiro de aplicação obtém um registro histórico de todas as mudanças feitas em cada nova versão do sistema. Essas mudanças são automaticamente incorporadas às novas versões geradas pelo GREN-Wizard, com o mínimo de intervenção do engenheiro de aplicação.

Outros estudos de caso devem ser realizados com a ferramenta para aprimorar os testes já realizados e adicionar melhorias, se for o caso. Além disso, é necessário utilizá-la em outros contextos, por exemplo, manutenção de software.

Apesar da ferramenta apresentada neste artigo ser específica a um determinado framework, infere-se a possibilidade de utilizar a sua modelagem conceitual e arquitetura para implementar outras ferramentas, com o mesmo propósito, em outros frameworks existentes.

Notou-se que a ferramenta *GREN-WizardVersionControl* é classificada como um sistema de SCM, pois atende a maioria das características desejáveis [20] para esse tipo de sistema: é fácil de utilizar, permite que o usuário utilize as funcionalidades da ferramenta como parte da execução de seu trabalho; é fácil de ser administrada; e é integrada com a ferramenta GREN-Wizard.

Referências

[1] P. Abrahamsson, O. Salo, J. Ronkainen, and J. Warsta. Agile software development methods. review and analysis. ESPOO (Technical Research Centre of Finland)' 2002. VTT Publications n. 478, 2002. `http://www.inf.vtt.fi/pdf/publications/2002/P478.pdf`. Acessado em: Dezembro,2003.

[2] K. Beck. *Extreme Programming Explained: Embrace Change*. Addison-Wesley, 2nd edition, 2000.

[3] R. T. V. Braga, F. S. R. Germano, and P. C. Masiero. A pattern language for business resource management. In *PLOP'1999, Conference on Pattern Languages of Programs*, pages 1–33, August 1999.

[4] R. T. V. Braga and P. C. Masiero. Building a Wizard for Framework Instantiation Based on a Pattern Language. In *OOIS'2003, 9th International Conference on Object-Oriented Information Systems*, pages 95–106, Geneva, Switzerland, September 2003. Lecture Notes on Computer Science, LNCS 2817.

[5] R.T.V. Braga. *Um Processo para Construção e Instanciação de Frameworks baseados em uma Linguagem de Padrões de Domínio Específico*. PhD thesis, Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2003.

[6] M. I. Cagnin, J. C. Maldonado, F. S. Germano, A. Chan, and R. D. Penteado. Um estudo de caso de reengenharia utilizando o processo PARFAIT. In *SDMS'2003, Simpósio de Desenvolvimento e Manutenção de Software da Marinha*, Niterói, RJ, 2003. CD-ROM.

[7] M. I. Cagnin, J. C. Maldonado, F. S. Germano, and R. D. Penteado. PARFAIT: Towards a framework-based agile reengineering process. In *ADC'2003, Agile Development Conference*, pages 22–31. IEEE, June 2003.

[8] Carnegie Mellon University. Capability maturity model integration - version 1.1. Technical Report CMU/SEI-2002-TR-003, Carnegie Mellon University, Software Engineering Institute, 2002.

[9] A. Chan, M. I. Cagnin, and J. C. Maldonado. Aplicação do processo de reengenharia PARFAIT em um sistema legado de controle de biblioteca. Documento de Trabalho, ICMC-USP, May 2003.

[10] Cincom. Visualworks 5i.4 non-commercial, 2003. `http://www.cincom.com/`. Acessado em: Fevereiro, 2003.

[11] CVS. Concurrent Versions System - The open standard for version control. `http://www.cvshome.org`. Acessado em: Dezembro, 2003.

[12] M. Fayad and D. C. Schmidt. Object-oriented application frameworks. *Communications of the ACM*, 40(10), 1997.

[13] M. Fowler and K. Scott. *UML Distilled - Applying the Standard Object Modeling Language*. Addison-Wesley, first edition, 1997.

[14] FreeVCS. Free Version Control System. `http://www.thensle.de`. Acessado em: Abril, 2004.

[15] IBM. ClearCase. `http://www-306.ibm.com/software/awdtools/clearcase`. Acessado em: Abril, 2004.

[16] ISO 9000. International Organization for Standardization. `http://www.iso.ch/iso/en/iso9000-14000/iso9000/iso9000index.html`. Acessado em: Abril, 2004.

[17] ISO/IEC 15504. International Standard for Software Process Assessment. `http://isospice.com/standard/tr15504.htm`. Acessado em: Abril, 2004.

[18] Ralph Johnson and Brian Foote. Designing reusable classes. *Journal of Object-Oriented Programming*, 1988.

[19] Microsoft. Visual SourceSafe. `http://www.msdn.microsoft.com/vstudio/previous/ssafe`. Acessado em: Abril, 2004.

[20] A. K. Midha. Software configuration management for the 21st century. *Bell Labs Technical Journal*, 2(1), 1997. `http://citeseer.nj.nec.com/midha97software.html`. Acessado em: Fevereiro, 2004.

[21] MySQL. Mysql reference manual, 2003. `http://www.mysql.com/doc/en/index.html`. Acessado em: Dezembro, 2003.

[22] M. C. Paulk. Capability maturity model for software - version 1.1. Technical Report CMU/SEI-1993-TR-24, Carnegie Mellon University, Software Engineering Institute, 1993.

[23] R. Pressman. *Software Engineering: A Practitioner's Approach*. McGraw-Hill, 5th edition, 2001.

[24] Quma Software, Inc. QVCS – Quma Version Control System. `http://www.qumasoft.com`. Acessado em: Abril, 2004.

[25] C. Reichenberger. Delta storage for arbitrary non-text files. *ACM*, pages 144–152, 1991.

[26] Synergex. PVCS. `http://www.pvcs.synergex.com`. Acessado em: Abril, 2004.

[27] Taligent. Building object-oriented frameworks, 1997. `http://www.ibm.com/java/education/oobuilding/index.html`. Acessado em: Fevereiro, 2002.

[28] Telelogic. Continuus/CM. `http://www.telelogic.com/continuus.cfm`. Acessado em: Abril, 2004.

[29] T. Tourwé. *Automated Support For Framework-Based Software Evolution*. PhD thesis, Department of Computer Science, Vrije Universiteit Brussel, Brussel, 2002.

[30] D. Turk, R. France, and B. Rumpe. Limitations of agile software processes. In *Third International Conference on Extreme Programming and Agile Processes in Software Engineering (XP'2002)*, pages 43–46, Alghero, Sardinia, Italy, May 2002.

Identificación de Señales Verbales en el Espacio de Fase Reconstruido

José A. Brito
Universidad de Los Andes, Postgrado en Computación,
Mérida, Venezuela, 5101
jabbmv@cantv.net

Wladimir J. Rodríguez
Universidad de Los Andes, Postgrado en Computación,
Mérida, Venezuela, 5101
wladimir@ula.ve

y

Flor E. Narciso
Universidad de Los Andes, Departamento de Computación,
Mérida, Venezuela, 5101
fnarciso@ula.ve

Abstract

In this paper we describe the use of Multilayer Perceptron Array for learning and classifying speech signals, using characteristic vectors of reconstructed dynamics. First, we consider the phonatory system as a black-box, where the only available data is its output: the speech signal. Theoretically, if reconstruction of system dynamics is properly made, geometric structures or attractors outlined in the space are topologically equivalent to original, and inaccessible, structures. This is a way of accessing underlying dynamics, and is the starting point for two kinds of experiments: classification of vowels and digits, with Venezuelan Spanish voices. Results verify positively that characteristics vectors extracted from underlying dynamics hold discriminative power for distinguishing between classes of speech signals. Besides, neural networks are able to generalize using this kind of data.
Keywords: Speech signals classification, reconstructed dynamics, pattern recognition, non linear dynamics, neural nets, SpeechDat.

Resumen

Este artículo se describe el uso de arreglos de redes neuronales de retropropagación para el aprendizaje y clasificación de señales verbales, usando vectores de características de la dinámica reconstruida. Primero, se considera el sistema fonatorio como una caja negra, donde la única data disponible es la salida: la señal verbal. Teoreticamente, si la reconstrucción de la dinámica del sistema es correcta, las estructuras geométricas o atractores del espacio son topologicamente equivalentes a las estructuras originales inaccesibles. Esta es una forma de acceder a la dinámica subyacente, y es el punto de partida para dos tipos de experimentos: clasificación de vocales y dígitos, con voces en español venezolano. Los resultados verifican positivamente que los vectores de características extraidos de la dinámica subyacente tiene poder discriminatorio para distinguir entre clases de señales verbales. Además, las redes neuronales son capaces de generalizar usando este tipo de datos.
Palabras claves: , Clasificación de señales verbales, espacio de fases reconstruido, reconocimiento de patrones, dinámica no lineal, redes neuronales, SpeechDat.

1. INTRODUCCIÓN

Recientemente, ha surgido el interés por la parametrización no lineal de señales verbales a partir de la reconstrucción de la dinámica del sistema fonatorio [5, 6]. En principio, se trata de resolver un problema de clasificación utilizando un enfoque cualitativo sobre perfiles del proceso físico subyacente [11, 12], en este caso, la fonación. Por el contrario, las técnicas convencionales de análisis recurren a la hipótesis de linealidad en la emisión verbal, aunque existen numerosas objeciones a dicho proceder. Por ejemplo, en el popular modelo **fuente-filtro** de generación de voz, la fuente de excitación es la turbulencia desarrollada en el mismo tracto vocal, por lo que en este caso la fuente natural no corresponde con el modelo. Además, un modelo lineal va a tener dificultades para ajustarse a la alta variabilidad de la señal. Por lo tanto, al final, estos modelos son solo una aproximación del sistema. El enfoque presentado aquí se basa en un arreglo de redes neuronales de retropropagación para el aprendizaje y clasificación de las señales verbales, usando vectores de características del espacio de fase reconstruido.

Se realizaron dos tipos de experimentos: vocales y dígitos. En ambos casos las señales fueron extraídas de la base de datos de voces venezolanas SpeechDat [4]. Por lo que este trabajo es el primero que emplean técnicas no lineales con voces venezolanas. En cada experimento se define dos corpus de voces: C_E y C_P, consistentes de señales de entrenamiento y prueba de la red neuronal respectivamente.

2. ESPACIO DE FASE RECONSTRUIDO

En el caso de los sistemas no lineales, con datos incompletos, la extracción de información nueva a partir de los datos resulta más difícil que en la contraparte lineal [2]. Si el sistema en cuestión es altamente complejo (eg., el sistema fonatorio), pero sólo una de sus propiedades (eg., la señal verbal) está al alcance de algún sensor, los procedimientos de análisis tradicionales resultarán muy limitados. Como alternativa, la reconstrucción del espacio de estados permite recuperar la dinámica de un sistema no lineal a partir de una única serie de tiempo [1]. En concreto, las trayectorias forman en este espacio unas estructuras geométricas denominadas atractores. Naturalmente, el espacio reconstruido no equivale completamente a la dinámica interna del sistema, pero bajo ciertas restricciones teóricas, preserva la topología de la misma. Esto permite que las conclusiones obtenidas en la dinámica reconstruida resulten válidas también en la verdadera e inaccesible dinámica interna (caja negra) [1, 7, 9]. Además, el espacio de fase reconstruido facilita la detección de estructuras que en la serie de tiempo podrían pasar desapercibidas.

A continuación se describe la obtención del espacio de fase reconstruido. Considérese un conjunto de muestras uniformemente espaciadas de una única variable, como la señal verbal S_v. El espacio de fase reconstruido es una representación multidimensional de la señal contra versiones demoradas de sí misma (subseries). En términos más formales, el espacio de fase reconstruido se forma mediante la definición de vectores $\mathbf{V}_n$ en $\Re^m$:

$$\mathbf{V}_n = \{S_v[n], S_v[n+\tau], ..., S_v[n + (m - 1)\,\tau]\} \qquad (1)$$

o

$$\mathbf{V}_n = \{S_v[n], S_v[n-\tau], ..., S_v[n - (m - 1)\,\tau]\} \qquad (2)$$

donde $S_v[i]$ es el valor de la señal en tiempo i (una muestra). Por su parte, m y τ son los parámetros fundamentales de reconstrucción conocidos como dimensión embebida y retardo respectivamente. El teorema de Takens [9], que relaciona el espacio de fase reconstruido con la verdadera dinámica interna del sistema, expresa que dados m y τ suficientes, la dinámica real y el espacio de fase reconstruido resultan topológicamente idénticos. Pruebas preliminares con el método de entropía diferencial [3] arrojaron valores bajos para m y τ sobre el corpus de vocales. De esta forma, se fijó $m = 2$ y $\tau = 3$ en los subsiguientes experimentos. La Figura 1 muestra el espacio de fase reconstruido para una vocal arbitraría, con una malla de 100 bloques sobre el plano. Por el otro lado, los dígitos constituyen señales muy complejas debido a su superior riqueza fonética, y consecuentemente, una representación espacial sencilla no es posible. Un enfoque algo diferente, discutido en la próxima sección, se utilizará para los dígitos.

Note que para cada eje en la figura corresponde al intervalo [-1,+1], esto se logra por medio de la normalización de la señal verbal: cada muestra se divide por $max(abs(S_v))$. Este paso, aunque trivial, es importante ya que permite que los bloques B_i sea de dimensión fija. Estrictamente, cada bloque es un cuadrado, y su área $4 / r$ (sin dimensiones), donde r es el total de bloques sobre el plano. En la figura 1, $r = 100$, y por lo tanto el área de cada bloque es 0.04.

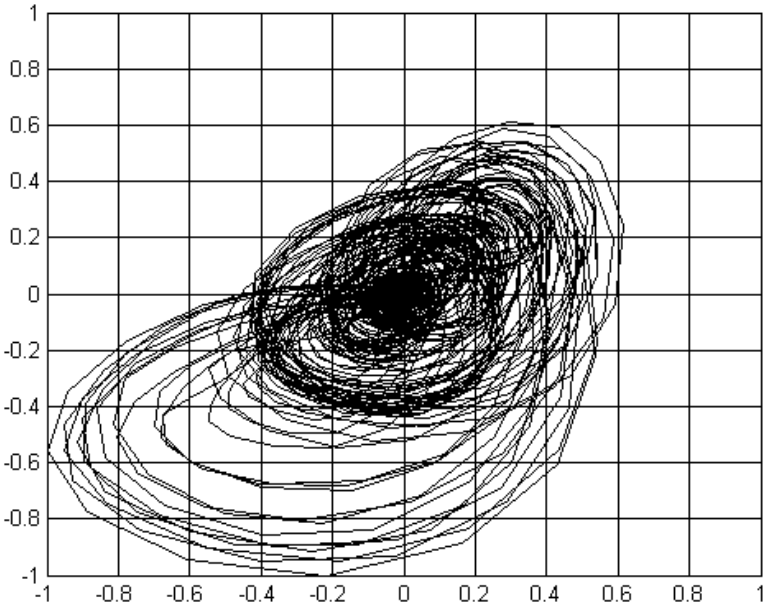


Figura 1. Espacio de fase reconstruido para una vocal en español venezolano

3. EXTRACCIÓN DE CARACTERÍSTICAS

Para las vocales, el vector de características V_C^V viene dado por la densidad espacial de cada bloque $B_i(1 <= i <= r)$. Así, V_C^V consta, en principio, de *r* elementos. En cada B_i la densidad espacial se computa como sigue:

$$spatialDensity(B_i) = \frac{|B_i|}{|S_v|} \qquad (3)$$

Adicionalmente, para dotar de robustez a la clasificación, V_C^V se extiende con los *r* elementos resultantes de aplicar el análisis anterior sobre la serie de tiempo $S_v^d = S_v[i + 1] - S_v[i]$. De esta forma, se incorpora al vector la rapidez de variación en la señal, mediante una aproximación con las primeras diferencias.

En el caso de los dígitos, el análisis descrito se aplica sobre tramas superpuestas proporcionadas, sin los *r* elementos de S_v^d. Para nuestros efectos, una trama es una secuencia de muestras, o en otras palabras, es una subsecuencia de S_v. En una señal dada, las tramas siempre poseen la misma longitud, pero entre señales, el tamaño de la trama puede variar. Precisamente se denominan proporcionadas porque su tamaño es una proporción del de la señal. La señal se divide en *np* segmentos de igual tamaño, *np* par. Siendo *LSeg* la longitud de cada uno de los *np* segmentos (es decir, *LSeg = Longitud(S_v) / np*), entonces la longitud de la trama *LTr* se establece en $2 \times LSeg$. La primera trama inicia con la primera muestra de la señal. De allí en adelante, una trama inicia en la muestra central de la trama que le antecede y se extiende *LTr* muestras. Por ende, las *np-1* tramas en una señal inician respectivamente en las muestras

$$i \text{ x } LSeg + 1 \ (0 \le i \le np - 2)$$

Posteriormente, se aplica en cada trama un análisis similar al de las vocales. De esta forma, el análisis de las tramas contribuye al vector de características de los dígitos, V_C^D con $(np-1) \times r$ elementos.

4. ARREGLO DE REDES PERCEPTRONICAS MULTICAPA

Sea *n* la cantidad de categorías a reconocer (en el caso de las vocales, *n=5*, y con los dígitos, *n=10*). Luego, para cada experimento, el clasificador consiste en un arreglo $[R_1R_2...R_n]$ de *n* redes perceptrónicas multicapa R_i. Así, se asocia una red con cada categoría. Una vez definido el corpus C_E, puede procederse con la sesión de entrenamiento del clasificador. Básicamente, R_i ($1 \le i \le n$) se entrena con todas las señales *j* ($1 \le j \le cardinalidad(C_E)$) pertenecientes a C_E. A todas las entradas de entrenamiento en las que se verifique *categoría(j) = categoría(i)*, se les asocia una salida igual a 1; en el otro caso, la salida es 0.

Posteriormente, al momento de clasificar, la señal de entrada se caracteriza y el vector resultante se administra a cada una de las *n* redes. La red con la salida más alta determina la categoría en la que se clasifica la señal.

Las redes neuronales usadas en el clasificador constan de tres capas. La cantidad de unidades en la entrada dependerá del tamaño del vector de características, sea V_C^V o bien V_C^D. Luego, si dicho vector incorpora *p* componentes, entonces se dispondrá de *p* unidades de entrada. Por ejemplo, si se particiona el espacio de estados en *100* bloques, y se calcula la densidad de puntos en cada uno, V_C^V, y por ende la capa de entrada de la red, constará de *200* elementos. Por su parte, en la capa oculta se ubican 5 unidades, y en la capa de salida se coloca una sola unidad. Las funciones de activación son sigmoides logarítmicas, a excepción de la neurona de salida, que emplea una función de transferencia lineal. Finalmente, el algoritmo usado en el entrenamiento de las redes es Levenberg-Marquardt [8]. Este es un algoritmo avanzado para la optimización no lineal, y suele converger al mínimo error más rápidamente que la retropropagación clásica, aunque su consumo de memoria resulta notoriamente elevado.

5. RESULTADOS

Para cada experimento a realizar, se construyó un MPA. Se estableció *r = 100*, y con los dígitos se fijó *np* en *6*. A fin de verificar el funcionamiento de los clasificadores, se les proporcionó como entrada las señales en los corpora de entrenamiento, obteniendo el 100% de exactitud en el reconocimiento. Ensayos dependientes del hablante arrojaron tasas promedio de reconocimiento del 92% y 65.5%. Por otro lado, con las pruebas independientes del hablante, más interesantes, se obtuvo, para las vocales, un reconocimiento promedio de 55% (ver tabla 1), y para los dígitos, una tasa de 66% (ver tabla 2).

Tabla 1. Matriz de confusión para vocales independientes del hablante

	a	**e**	**i**	**o**	**u**	**%**
a	8	2	2	8	0	40.00
e	0	8	9	3	0	40.00
i	0	4	12	2	0	60.00
o	2	2	0	14	2	70.00
u	0	1	3	3	13	65.00
						55.00

Tabla 2. Matriz de confusión para dígitos independientes del hablante.

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**%**
0	11	0	0	0	0	2	0	7	0	0	55.00
1	0	12	2	4	0	0	0	0	1	1	60.00
2	0	0	13	3	1	0	0	0	0	3	65.00
3	2	0	2	14	0	1	0	1	0	0	70.00
4	1	0	0	0	17	0	0	2	0	0	85.00
5	1	0	0	0	0	11	0	8	0	0	55.00
6	2	1	0	4	1	0	9	1	0	2	45.00
7	3	0	0	0	1	2	0	14	0	0	70.00
8	0	0	1	0	1	0	0	0	18	0	90.00
9	0	3	0	2	0	1	0	1	0	13	65.00
											66.00

Obsérvese cómo los valores en las diagonales principales de las matrices de confusión confirman la tendencia de los MPA a clasificar correctamente las señales de entrada. En el caso de las vocales, la variabilidad entre las señales independientes del hablante deteriora la exactitud de reconocimiento. De forma interesante, en los dígitos no acontece así, quizás porque estas señales incluyen más información que las vocales, y el MPA logra capturarla.

Algunos estudios han abordado previamente la caracterización de la densidad de los atractores en el espacio de estados, para clasificar señales verbales. Por ejemplo, en [10] se emplea un clasificador bayesiano, con una exactitud promedio de 34.49% en las vocales, si bien se trata de vocales inglesas. En [5, 6] se utiliza el espacio de información difuso, con un reconocimiento perfecto, pero con un corpus de sólo seis señales. Comparando con técnicas en el dominio de la frecuencia, se tiene el trabajo de Maldonado [4], el cual procede sobre corpus de dígitos venezolanos, con tasas de reconocimiento superiores al 90%. Empero, este último trabajo utiliza aproximaciones analíticas ya establecidas, como coeficientes ceptrales y modelos ocultos de Markov.

6. CONCLUSIONES

Considerando que este análisis esta basado completamente en el dominio del tiempo, las ratas de reconocimiento son bastantes buenas. Sin embargo, se necesita realizar más investigaciones para determinar el efecto de un análisis dimensional mayor $m > 5$. Cuando se utilizan más de dos dimensiones, la caracterización se hace más difícil. Excepto por el entrenamiento de las redes neuronales, las técnicas expuestas no requieren de recursos computacionales muy altos. Por lo que habría que preguntarse si un análisis de altas dimensiones es conveniente, tomando en cuenta la precisión de las técnicas en el dominio de frecuencias. Este tipo de investigación es necesaria debido que el aprendizaje de más fenómenos seguramente necesitaran más atractores.

Finalmente, existe una información discriminante en la dinámica reconstruida, y las redes neuronales, son capaces de generalizar lo que distingue entre las clases de señales. Se encontró que la conjunción entre las características de la dinámica reconstruida y las redes neuronales contiene el suficiente poder discriminatorio para clasificar las señales de voz, aunque es necesario continuar con las investigaciones.

Referencias

1. H. Abarbanel, R. Brown, J. Sidorowich, y L. Tsimring, "The analysis of observed chaotic data in physical systems", *Reviews of Modern Physics*, vol. 65, No. 4, 1993.
2. E. Bradley, "Time series analysis", in *Intelligent Data Analysis: An Introduction*, Springer, 1999.
3. T. Gautama, D. Mandic, y M. Van Hulle, "A differential entropy based method for determining the optimal embedding parameters of a signal", *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, 2003.
4. J. L. Maldonado, *Tratamiento y reconocimiento automático de señales de la voz venezolana*, Disertación doctoral, Universidad de Los Andes, 2003.
5. W. Rodríguez, H.-N. Teodorescu, F. Grigoras, A. Kandel y H. Bunke, "A fuzzy information space approach to speech signal non-linear analysis", *International Journal of Intelligent Systems*, vol. 15, No. 4, pp. 343-363, 2000.
6. W. Rodríguez, "Similarity of Dynamical Systems", Ph.D. Thesis, University of South Florida, 1998.
7. T. Sauer, J. A. Yorke, y M. Casdagli, "Embedology", *Journal of Statistical Physics*, vol. 65, pp. 579-616, 1991.
8. Shepherd, A. J. *Second-Order Methods for Neural Networks*, Springer-Verlag, 1997.
9. F. Takens, "Detecting strange attractors in turbulence", *Dynamical Systems and Turbulence*, Warwick, 1980.
10. J. Ye, M. T. Johnson, y R. J. Povinelli, "Phoneme Classification using Naive Bayes Classifier in Reconstructed Phase Space", *10th IEEE Digital Signal Processing Workshop*, 2002.
11. F. Zhao, "Extracting and Representing Qualitative Behaviors of Complex Systems in Phase Space", *Artificial Intelligence*, vol. 69, pp. 51-92, 1994.

Diseño de un Medio de Gestión de Servicios para Sistemas Multiagentes

Víctor R. Bravo
Postgrado de Computación. Facultad de Ingeniería
Universidad de los Andes
Mérida – Venezuela.
victorb@cptm.ula.ve

y

José L. Aguilar, Franklin Rivas, Mariela Cerrada
Centro de Microcomputación y Sistemas Distribuidos (CEMSID)
Escuela de Ingeniería de Sistemas, Facultad de Ingeniería
Universidad de los Andes
Mérida – Venezuela
{aguilar, rivas}@ula.ve

Abstract

Agent based programming is a new paradigm to build software systems. It is based on the generation of software modules with capacities like communication and autonomy on its actions that facilitates the construction of auto-organized and more efficient complex systems. In this paper a middleware based on agents for a computational platform of MultiAgents Systems is developed. Particularly, the middleware has been used like one of the components of the platform SCDIA (Distributed Control System based on agents). This middleware provides services for access and administration of hardware resources, applications, data and agents, and it has qualities associated to distributed systems such as interoperability, migration, security, naming, communication, among others.
Keywords: Middleware, Intelligent Systems, MultiAgent Systems, Software engineering, knowledge engineering.

Resumen

La programación basada en agentes constituye un nuevo paradigma en la construcción de sistemas de software. Ella se basa en la generación de módulos de software que tengan capacidades de comunicación y autonomía sobre sus acciones, lo que posibilita la construcción de sistemas complejos autorregulados y más eficientes. En este trabajo se desarrolla un Medio de Gestión de Servicios (MGS) basado en agentes que formaría parte de una plataforma computacional para ejecutar Sistemas Multiagentes. Particularmente, el MGS ha sido usado como uno de los componentes de la plataforma SCDIA (Sistema de Control Distribuido basada en agentes) propuesta en [2,5]. Este medio proporciona servicios de acceso y gestión de recursos de hardware, de aplicaciones, de datos y de agentes, y posee las cualidades asociadas a los sistemas distribuidos tales como interoperabilidad, migración, seguridad, nombramiento, comunicación, entre otras.
Palabras claves: Medio de Gestión de Servicios, Sistemas Inteligentes, Sistemas Multiagentes, Ingeniería de Software, Ingeniería de conocimiento.

1. INTRODUCCION

Los sistemas multiagentes (SMAs) son sistemas de software que agrupan un conjunto de cualidades pertenecientes a los tópicos más avanzados que en el área de computación se tratan en la actualidad. Esta conjunción de conceptos y tecnologías, hace posible dar respuestas a problemas complejos, a través de la construcción de sistemas auto-organizados y dinámicos, que pueden integrarse con aplicaciones legadas (Las aplicaciones legadas son aplicaciones operativas que cuentan con una utilidad real, por lo tanto es inconveniente su reemplazo o desecho), y sistemas heterogéneos.

El paradigma orientado a agentes tiene años de estudio e investigación en los espacios académicos, y se han propuesto una gran cantidad de modelos y prototipos de aplicaciones que dan muestra de su utilidad [3,4]. El paradigma no se ha establecido en una gran mayoría de ambientes empresariales e industriales, a excepción del sector de telecomunicaciones donde se han desarrollado una gran cantidad de aplicaciones que actualmente funcionan y resuelven una gran cantidad de problemas utilizando teorías de agentes [3].

Se percibe una creciente actividad sobre este tema en los departamentos de investigación y desarrollo de importantes empresas del ramo del software, tal es el caso de IBM (International Business Machines), Microsoft, HP, entre otras. Además, la existencia de organizaciones tales como FIPA (Foundation Intelligent Physical Agents, organización que agrupa importantes empresas del campo tecnológico que han dictado un conjunto de especificaciones para la construcción de sistemas multiagentes), evidencia que este tipo de tecnología ofrecerá mejores y nuevas soluciones a problemas que actualmente enfrentan las organizaciones que gestionan y mantienen procesos complejos.

Por otro lado, las tendencias actuales en automatización de procesos continuos están orientadas al desarrollo de sistemas de control basados en una arquitectura jerárquica de referencia. Atendiendo a estas tendencias se ha definido un Modelo de Referencia para el Desarrollo de Sistemas de Control Distribuidos Inteligentes basado en Agentes (SCDIA), representado por una jerarquía de cinco niveles, entre los cuales se distribuye la información del sistema y las tareas de control [1, 2].

La adaptación de SMAs dentro de plataformas tecnológicas ya operativas, no puede considerarse como un problema trivial. La heterogeneidad trae como consecuencia un extenso y difícil trabajo de acoplamiento de interfaces; aunado a esto, el nuevo paradigma basado en agentes trae consigo nuevos conceptos que deben ajustarse a los ya existentes. Por ejemplo, cuando se habla de "conversaciones" entre agentes, es necesario trasladar este concepto al contexto de computación distribuida en el cuál se manipulan objetos, mensajes, protocolos, procesos, con la finalidad de implantar la comunicación entre entes que llamaremos agentes. La necesidad de contar con un conjunto de servicios que brinden conexión con recursos y aplicaciones puede constituirse en una herramienta indispensable para la integración del paradigma de agentes a las plataformas computacionales de las organizaciones de hoy en día. En este proyecto se diseña e implementa un Medio de Gestión de servicios (middleware) que permite la comunicación entre recursos, aplicaciones, y agentes. En nuestra propuesta se hace uso de conceptos tales como componentes, reflexión y computación distribuida; que brindan la base teórica para el trabajo planteado.

2. ASPECTOS TEÓRICOS

2.1 Sistemas Multiagentes (SMAs)

Cuando se habla de agentes en computación no existe un consenso sobre su definición. El concepto de "agente" tiene asociado un conjunto de cualidades, que, aunque cada una de ellas tiene una definición clara, no tienen ni una única interpretación ni una sola forma de implementarse. Ahora bien, un agente debe reunir algunas de las siguientes características:

- *Autonomía:* los agentes son autónomos en la medida en que actúan sin la intervención humana ni de otros sistemas externos [12]. Se puede dar una noción general de autonomía, definiéndola como la capacidad que tiene un agente de tener un comportamiento propio, y reaccionar a los estímulos externos basándose en su estado interno.
- *Comunicación:* la comunicación entre agentes tiene entre sus objetivos acercarse a lo que es la comunicación entre personas, a través de intervenciones dinámicas con oraciones y frases que contienen significados. La capacidad de cada agente de conversar utilizando un lenguaje basado en ontologías y realizar intervenciones asíncronas, constituye un paso adelante en llevar el concepto real de conversación al ámbito computacional.
- *Sociabilidad:* esta capacidad está muy relacionada con el aspecto comunicacional. Una sociedad de agentes es un grupo de agente que interactúan, se comunican, conversan, "piensan", y actúan en conjunto para lograr un objetivo común.
- *Reactividad:* Es la capacidad de emitir una acción inmediata al recibir una señal o percibir un estado en el ambiente.
- *Inteligencia:* se pueden utilizar un conjunto variado de técnicas inteligentes en la construcción de agentes, entre las cuales se pueden nombrar las reglas difusas para analizar situaciones dinámicas, las redes neuronales para predecir comportamientos y variables del ambiente, las colonias de hormigas como una técnica de coordinación entre agentes, los algoritmos genéticos como método de búsqueda, entre otros. En la medida en que se integren más técnicas inteligentes a la construcción de agentes, se logrará un acercamiento más certero a la característica inteligente de los agentes.

- Movilidad: *es la capacidad que tiene un agente de mover su estado y código de ejecución de un nodo a otro en un sistema distribuido.*

No existe una única arquitectura para diseñar y construir agentes, por ejemplo, una corriente se ha inclinado por el diseño de agentes utilizando programación lógica, lo que supone una estrecha vinculación a un motor de inferencia con colecciones de hechos; otra corriente ha vinculado los agentes con los sistemas de control, y también, existen arquitecturas específicas basadas en creencias y deseos.

2.2 Sistemas de Control Distribuidos Inteligentes basados en Agentes

Los Sistemas de Control Distribuidos Inteligentes basados en agentes (SCDIA) es específicamente una plataforma de SMA diseñada para sistemas de automatización industrial [1, 2]. Propone una colección de agentes caracterizando los elementos de control de procesos y definiendo un mecanismo genérico para el manejo de las actividades organizadas relacionadas con la automatización industrial. Así, los agentes del SCDIA son:

- Agente Medición: recoge la información necesaria para obtener el estado del proceso.
- Agente Controlador: toma acciones basadas en el estado del sistema.
- Agente Coordinador: modifica las decisiones del agente controlador y establece nuevos objetivos y servicios. Coordina la comunidad de agentes.
- Agente Actuador: ejecuta las decisiones tomadas por el agente controlador, agente coordinador, y/o agentes especializados.
- Agentes Especializados: ellos ejecutan las tareas especiales de la comunidad de los agentes.

El SCDIA es dividido en dos niveles: un nivel de la interacción con el ambiente dónde el agente medición y el agente actuador viven; y un nivel de decisión donde los otros agentes de la comunidad viven (ver figura 1). Nosotros llamamos a este conjunto de cinco agentes, la Comunidad de Agentes de Control.

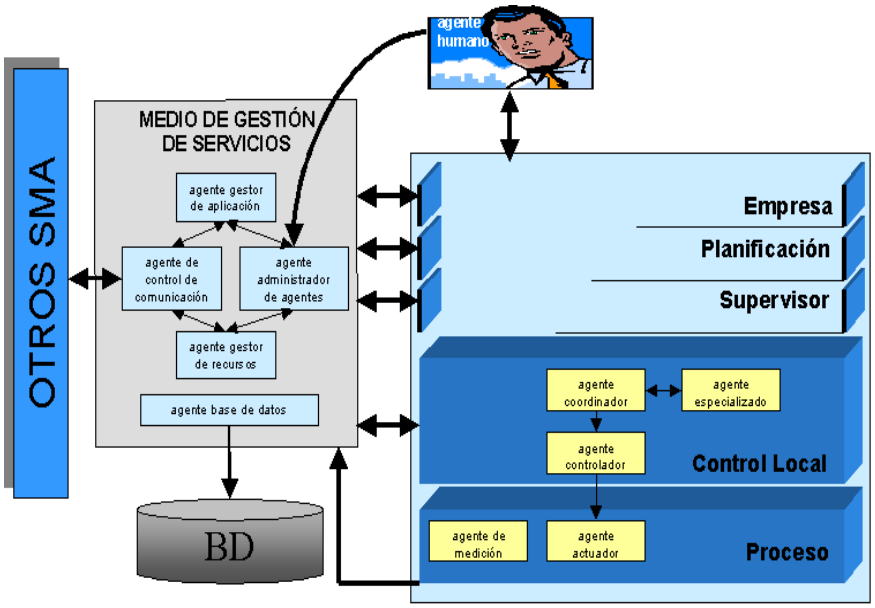


Figure 1. Modelo SCDIA

El SCDIA también propone una comunidad de agentes que manejan a los Agentes de Control. Esta comunidad se llama el Medio de Gestión de Servicios (MGS) [18]. El MGS permite la migración, la localización, la activación/desactivación de agentes, y el conocimiento del estado global del SCDIA; todo estos trabajos se llevan a cabo por el Agentes del MGS. Otro trabajo que el MGS realiza es controlar el inventario de todas las aplicaciones y recursos que se manejan y/o se proporcionan por los agentes. Por otro lado, también se necesita la administración de los datos guardados dentro del sistema; este trabajo lo lleva a cabo el Agente de la Base de datos del MGS. Finalmente, el SCDIA tiene la capacidad para comunicarse con otros SMA a través del MGS, específicamente por el Agente de Control de Comunicaciones.

2.3. Componentes

Los componentes son piezas de software reutilizables, disponibles para diferentes lenguajes y aplicaciones a través de una interfaz común. Permiten encapsular el acceso a datos, la lógica de negocio, el acceso a recursos de hardware

y de software, entre otras cosas. Se basan en una extensión de los objetos, tienen métodos y propiedades, y permiten persistencia, acceso remoto y un modelo distribuido.
En la actualidad existen varios esquemas que plantean una visión orientada a componentes Los dos esquemas más utilizados son el COM (Component Object Model) [22], propuestos para sistemas bajo sistemas operativos Windows, y el modelo basado en Beans, propuesto por Sun Microsystems para arquitecturas basadas en tecnología Java.

3. PROPUESTA DEL MEDIO DE GESTIÓN DE SERVICIOS (MIDDLEWARE)

Se desarrolló una capa intermedia de Gestión de Servicios (Middleware) constituida por componentes de software en un ambiente distribuido y heterogéneo. Cada componente puede actuar como vía de acceso al procesamiento para una determinada aplicación, como puente entre clientes remotos y fuentes de datos, o interfaz de acceso a recursos y sistemas de información. El middleware es el corazón del sistema de agentes distribuido, puesto que en él moran los agentes que manejan los servicios de comunicación y le otorgan al sistema de agentes características tales como seguridad, transparencia, nombramiento, migración e interoperabilidad. El modelo de middleware está estructurado en tres niveles (ver figura 2).

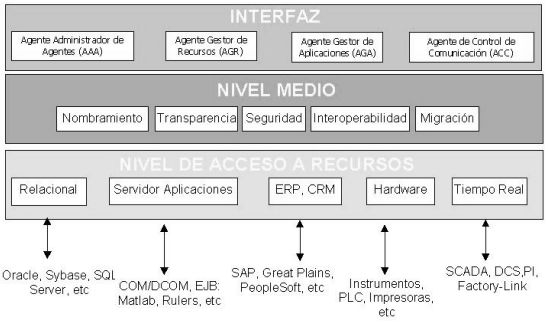


Figura 2. Arquitectura del MGS

Nivel Interfaz: es el nivel de interacción con agentes y usuarios. A esta capa pertenecen cinco agentes del SCDIA: Agente Administrador de Agentes (AAA), Agente Gestor de Recursos (AGA), Agente Gestor de Aplicaciones (AGA), Agente de Base de Datos (ABD) y Agente de Control de Comunicación (ACC). Estos interactúan con el nivel medio a través de una interfaz que permite la comunicación entre los dos niveles. El nivel interfaz es el encargado de establecer las pautas de conversación entre los componentes del sistema y los agentes, como también, definir los esquemas de coordinación con los agentes de nivel superior (por ejemplo, con la comunidad de agentes de control del SCDIA).

Nivel Medio: constituye el núcleo del sistema distribuido, provee servicios al nivel interfaz. Garantiza transparencia y seguridad en las transacciones, interoperabilidad de las aplicaciones y componentes de software, migración de agentes, objetos y/o recursos, comunicación ínter-proceso, y provee un sistema de nombramiento para la localización de agentes y/o objetos.

Nivel de Acceso a Recursos: Esta capa está integrada por todos los manejadores asociados a recursos directos, tales como, manejadores para acceso a datos relaciones, que implementan conexiones ODBC, JDBC, ADO, DAO, ADO.NET; manejadores de acceso a sistemas en tiempo real y a sistemas supervisores, acceso a sistema de planeación empresarial y gestión de recursos (ERP: Enterprise Resources Planning), acceso a sistemas de gestión de relaciones con los clientes (CRM); acceso a aplicaciones o componentes de software; y acceso directo a hardware específico.

3.1 Características Nivel Interfaz

Como se dijo antes, esta capa esta compuesta por los agentes que sirven de interfaz entre el MGS y la comunidad de agentes de control del SCDIA. Esta capa está compuesta por los siguientes agentes:

Nombre: Administrador de Agentes (AAA).
Tipo: Agente de Software.
Papel: Administrador del sistema multiagentes.
Descripción: se encarga de manejar, integrar y supervisar el estado del sistema multiagente. Este agente conoce la localización y estado de todos los agentes que existan en el sistema. El AAA dirige las migraciones de los agentes; así, cada agente que se mueve de un nodo a otro debe notificar al AAA el movimiento que ha efectuado; de manera que el agente administrador siempre tenga una vista ajustada al estado del sistema en tiempo real

Nombre: Base de Datos (ABD).
Tipo: Agente de Software.

Papel: Gestor de Datos en el SCDIA.
Descripción: este agente se encarga de establecer el enlace con los lugares donde existan datos de interés para el proceso que se esté ejecutando, sea que estos datos provengan de bases de datos (relacionales, orientadas a objetos, tiempo real, etc.), de SCADAS, DCS, medidores, o cualquier otro dispositivo o aplicación que pueda almacenar datos. Además, el agente debe permitir el traslado de los datos entre los diferentes dispositivos y/o aplicaciones de una manera transparente. Responde a las peticiones de los agentes de bases de datos, agentes de aplicaciones, agentes de recursos, y los agentes de la comunidad de agentes de control del SCDIA.

Nombre: Gestor de Aplicaciones (AGA).
Tipo: Agente de Software.
Papel: Administrador de aplicaciones del SCDIA.
Descripción: este agente se encarga de ubicar las aplicaciones que puedan ser requeridas por un proceso que se esté ejecutando, como por ejemplo de acceso a redes, programas de cálculo numérico o simbólico, aplicaciones de inteligencia artificial, de envío y recepción de mensajes, etc. Dichas aplicaciones pueden estar en cualquier servidor al que se tenga acceso y son requeridas por los agentes de bases de datos, de administración de recursos, y algunos de los agentes de la comunidad de agentes de control del SCDIA (coordinadores y especializados).

Nombre: Gestor de Recursos (AGR).
Tipo: Agente de Software
Papel: Administrador de recursos de Hardware del SCDIA
Descripción: este agente se encarga de distribuir el uso de los dispositivos (hardware) necesarios en la ejecución de un proceso, como por ejemplo procesadores, dispositivos de entrada/salida, dispositivos de almacenamiento, etc. Este agente puede ser accedido por cualquier agente del SCDIA.

Nombre: Control de Comunicación (ACC).
Tipo: Agente de Software.
Papel: Administrador de comunicaciones del SCDIA.
Descripción: es el encargado de mantener y controlar la comunicación entre sistemas multiagentes. Se encarga de traducir y manipular ontologías, y mantener un estado confiable del canal de comunicación.

3.2 Características del Nivel Medio

En cuanto al nivel medio, este le proporciona ciertas funcionalidades al MGS, tales como:

- *Transparencia:* se refiere al manejo de servicios siguiendo la filosofía de ocultar detalles de "cómo" se provee el servicio concentrándose en "que" provee el servicio, es decir, al agente se le ocultan detalles sobre la ejecución del servicio que no se consideran relevantes. Los servicios se proveen manejando en una capa oculta los detalles de implementación. La transparencia facilita la implantación de un sistema multiagente en una plataforma heterogénea. Para lograr transparencia, se debe utilizar una interfaz uniforme entre componentes, usuarios y agentes. En ciertos casos en necesario utilizar técnicas de envoltorio (wrapper), computación reflexiva, integración de código (.NET), con la finalidad de normalizar la interacción entre los componentes del sistema multiagente.

- *Seguridad:* debido a que el esquema distribuido opera bajo un sistema en red y de acceso externo, es necesario implementar mecanismos para mantener un nivel de seguridad que imposibiliten el acceso a aplicaciones y a datos por una fuente no acreditada. Cada agente debe ser autentificado cuando ingrese a un nodo de la plataforma, es decir, el nodo que acepta el agente debe saber el perfil del agente, su propietario, permisos con los que cuenta, y tareas que desempeña. En el mercado existen diferentes y variadas aplicaciones que implementan estos mecanismos, entre estos mecanismos podemos citar los algoritmos que utilizan criptografía, los algoritmos de validación de contraseñas, y los algoritmos de clave pública y privada, entre otros.

- *Interoperabilidad:* un problema recurrente en sistemas computacionales en red es el de integrar diferentes plataformas de software, diferentes aplicaciones corriendo en sistemas operativos diferentes que deben cumplir una función en conjunto, etc. La aparición de nuevos lenguajes de programación que implementan máquinas virtuales o código 100% portable, sistemas basados en objetos como CORBA, y métodos menos restrictivos como SOAP (**S**imple **O**bject **A**ccess **P**rotocol), han dado una respuesta a este problema.

- *Migración:* en sistemas basados en tecnología Java o CORBA, u otra tecnología similar, es posible implantar agentes que migran o que se mueven entre los nodos de una red. El agente administrador de agentes es el encargado de controlar el proceso de migración. Existen varias técnicas y modos de migración para agentes y objetos. Para lograr migración de agentes es necesario contar con una plataforma comunicacional con ciertas características, por lo general esta plataforma debe permitir el uso del protocolo TCP/IP. Los agentes móviles, generalmente se utilizan para recolectar información de varios servidores; se mueven o migran utilizando un protocolo común, y establecen sus operaciones localmente. Las plataformas distribuidas proveen herramientas para migración de objetos, estas herramientas pueden ser utilizadas en el proceso de migración de agentes. En el caso de los sistemas multi-agentes se proveen canales que permiten que los agentes viajen a través de los

diferentes nodos de la red. Es este tipo de ambiente un elemento importante a proveer es la seguridad, ya que una incursión de un agente o software externo puede reportar graves daños.

- *Mensajería y comunicación interprocesos*: es posible comunicar aplicaciones que corren en localidades diferentes a través de sistemas de mensajerías. Las aplicaciones pueden comunicarse entre ellas a bajo nivel, por ejemplo, a través de sockets (bajo protocolo TCP/IP), y a niveles más altos con lenguajes de marcas como XML (e**X**tended **M**arkup **L**anguage). También existen otros mecanismos de comunicación que son provistos por lenguajes o plataformas como Java o CORBA. Sistemas de pases de mensajes o de pizarrón (memoria compartida) también son aplicables. Todos estos sistemas de mensajería sirven de apoyo a lenguajes de alto nivel provistos para comunicación entre agentes, tales como KQML o ACL [12].

- *Sistema de Nombramiento:* para poder localizar eficientemente agentes, objetos y/o recursos es necesario contar con un sistema de nombramiento confiable que indique la función y asignación del agente u objeto en cuestión. También debe permitir acceder y manipular estos agentes u objetos de forma no ambigua. El sistema de nombramiento, también llamado "páginas blancas", debe ser administrado por un agente administrador. Este agente tiene la potestad de crear, iniciar, suspender y autorizar migraciones de todos los agentes. Los sistemas de nombramiento se implantan a través del uso de una base de datos que asocia recursos con nombres descriptivos, a los cuales se les puede asociar una jerarquía. Uno de los ejemplos de sistemas de nombramiento es el DNS (Domain Naming Server); sistema encargado de dirigir el nombramiento en la Internet y que implementa "espacios de nombres*" (namespace) o "dominios de nombres*", para soportar dominios de localidad por función o por servicio.

3.3 Características del Nivel de Acceso a Recursos

Un esquema para acceso a recursos integra los componentes y recursos al sistema multiagente. El esquema se realiza implementado un traductor o proxy que comunica el recurso con los otros componentes del sistema. El traductor hace uso del concepto de reflexión. La reflexión se refiere a la capacidad de un programa de manipular en tiempo de ejecución objetos, métodos, parámetros y variables propias, es decir, la propiedad de manipular los elementos que lo componen. También es posible, a través de la reflexión manipular los elementos de otras aplicaciones, de tal manera, que en tiempo de ejecución se conocen nombres, tipos y se pueden ejecutar métodos y funciones sin conocer previamente su nombre, tipo de valor devuelto, cantidad o los tipos de parámetros.
El traductor evalúa y compara los objetos del tipo "conceptos" y "acciones" que se tratan en las conversaciones de los agentes con los componentes donde se encapsulan los recursos, y realiza una sincronización entre ellos. Esta sincronización se realiza en tiempo de ejecución, cuando se produce un "acto de habla", un agente recibe un mensaje contentivo de los elementos ontológicos (concepto, acción, etc.), los evalúa y sincroniza con el componente indicado, ejecutando la acción descrita por el acto de habla.
Para encapsular los recursos es posible utilizar tecnologías como DCOM/COM (Distributed/Component Object Model), Java Beans, JNI (Java Native Interface), CORBA y XML (eXtended Meta Language), que permiten acceder al recurso exportando su interfaz a un lenguaje de definición común.
Así, en este nivel se define un esquema para acceder y lograr la comunicación con todos los recursos ubicados en las diferentes plataformas computacionales, de una manera estándar y transparente.

4. IMPLEMENTACIÓN

Para la fase de implementación se siguieron los requerimientos planteados en el diseño. Estos requerimientos están diseñados en función de las características fundamentales de los agentes, garantizando funcionalidades inherentes a los sistemas distribuidos. En este sentido se utiliza JADE[4] (Java Agent Development Enviroment), el cual es un entorno de desarrollo para Sistemas Multiagentes escrito en Java. Permite comunicación bajo el modelo RMI y IIOP-CORBA. Fue desarrollado por el TILAB (Telecom Italia Laboratory) y la Universidad de Parma, Italia. Se distribuye bajo licencia LGPL (Lesser General Public License)[1]. JADE ofrece un conjunto de bibliotecas de clases, que ofrecen servicios y funcionalidades para la construcción y administración de sistemas multiagentes. Entre las capacidades de la plataforma está la característica de interoperabilidad, heredada de Java, que permite acceder a plataformas heterogéneas. También ofrece clases para la gestión de ontologías y de comunicación, sistema de nombramiento, y de administración de agentes, y otras características importantes inherentes a los sistemas multiagentes.
JADE cumple con las especificaciones FIPA, en consecuencia se heredan las características de esta plataforma. Se implementa un modelo de comunicación basado en el lenguaje ACL; que utiliza ontologías y pase de mensajes para establecer conversaciones entre agentes. Además, es posible utilizar CORBA y protocolos para la Internet como el SMTP y HTTP.
Para la implementación de los agentes del MGS se utiliza un modelo basado en comportamientos. Los comportamientos son lógicas de acciones a seguir que pueden catalogarse en secuenciales, cíclicas, compuestas, paralelas, basadas en máquinas de estado, entre otras. Para cada agente se establece un tipo comportamiento,

[1] Esta tipo de licencia para código abierto puede accederse en el sitio web http://www.gnu.org

ejecutado en función de las variables de ambiente y la comunicación que el agente establezca con los demás agentes del sistema.
Los agentes AAA, AGA y AGR constituyen el núcleo del medio de servicios, ya que estos agentes controlan, supervisan, y modifican el estado global del sistema multiagente. Existen dos importantes servicios que brindan estos tres tipos de agentes, el primero es el control general de los agentes, que incluye creación, iniciación, migración, suspensión, destrucción, reactivación, clonación, nombramiento y búsqueda de agentes en todo el sistema. Estas tareas son coordinadas por el AAA, por lo cual todo agente debe tener comunicación con el administrador.
El segundo servicio es el de acceso a recursos de software y hardware, este servicio es administrado por los agentes AGA y AGR, y provee una interfaz basada en comunicación ACL (Agent Comunication Language). Para eso, se tiene una base de datos de componentes que envuelven el acceso a aplicaciones y recursos de hardware. Cuando es necesario integrar una aplicación se construye un componente J2EE (Java 2 Enterprise Edition) o COM y se agrega a la base de datos. El AGA posee una ontología para esta base de componentes que es compartida con los demás agentes. Para extender el acceso a recursos de hardware y de software se utiliza una interfaz JNI (Java Native Interface), que permite el acceso a componentes COM [11].
Para el acceso a recursos de hardware se cuenta con el AGR. Para cada nodo del sistema se tiene un AGR local que tiene el inventario de los recursos locales, además de que es quien administra y aplica algoritmos de balanceo de carga. Este agente posee una ontología que es utilizada para entenderse con los demás agentes del sistema.
El agente de base datos provee el servicio de acceso a datos. Los datos pueden ser accedidos desde distintas fuentes: servidores de base de datos, archivos de base de datos, archivos de texto o de hojas de cálculo, registros del sistema, etc. Para proveer heterogeneidad se utilizan los estándares JDBC para sistemas Java, y ADO para sistemas Windows.
Por último, el agente de control de comunicación es el encargado de establecer comunicación con otros sistemas multiagentes, posee una serie de protocolos que utiliza para traducir y establecer conversaciones.

5. CASO DE ESTUDIO

Para mostrar las características del MGS se utiliza una aplicación de optimización, que generalmente está incluida dentro del rango de aplicaciones de una plataforma de automatización.

5.1 Descripción del problema

En ambientes industriales, en ocasiones es necesario identificar procesos físicos que se realizan a diario en alguna de las plantas del complejo. Debido a que es difícil elaborar un modelo matemático exacto o aproximado siguiendo las leyes físicas del proceso, generalmente se plantea la construcción de un modelo basado en redes neuronales [3,5], que utilice datos históricos del proceso estudiado. Se tomó como ejemplo un reactor biológico, para el cual se tiene un conjunto de datos históricos de entrada y salida. Dado que uno de los objetivos del MGS es administrar eficientemente los recursos y aplicaciones de una plataforma de automatización, se toma ésta aplicación y se implementa utilizando el MGS. Para el ejemplo tratado se escogió una red neuronal de retropropagación[3,5]. La interfaz de usuario de está aplicación es un agente, por lo tanto posee todas las cualidades de los agentes del MGS. Al usuario se le muestra un cuadro de diálogo para que ingrese la ruta del archivo que contiene la configuración de la red neuronal y los archivos de datos (ver figura 3).

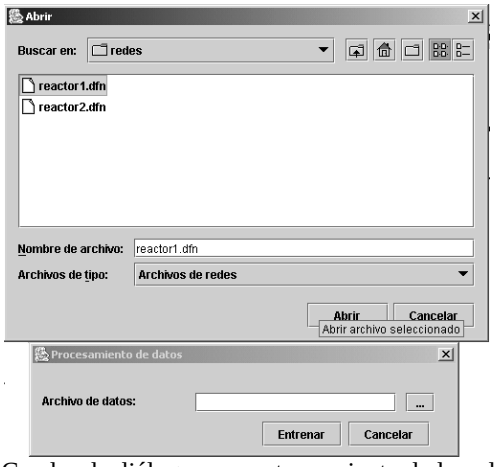


Figura 3. Cuadro de diálogo para entrenamiento de la red neuronal

En el archivo de configuración se especifican los nombres y tipos de variables de entrada y salida, además de los parámetros de entrenamiento, y en el archivo de datos se especifican en filas los patrones para entrenar la red:

```
// proceso.dfn
continuous entrada1u
continuous saliday
coef  0.1
momen 0.7

// proceso.dat    -------> Data
1.2 3
2.1 5
6.7 6
```

En el archivo de configuración de la red también se escriben los parámetros del entrenamiento, tales como el coeficiente de aprendizaje (*coef*) y el momento (*momen*). El coeficiente de aprendizaje es un valor numérico que ajusta el grado en que los pesos son modificados, por lo tanto, se dice que en este mismo grado la red aprende, por su parte, el parámetro "momento" permite a la red hallar en teoría, mejores errores.

5.2 Funcionamiento del MGS

Al presionar el botón "Aceptar", se inicia el proceso de gestión distribuida para el procesamiento de la información recolectada. En este proceso participan los agentes AGR, AAA; AGA, los agentes de la comunidad de agentes de control del SCDIA, y los recursos y aplicaciones que sean necesarios y estén disponibles para lograr los resultados esperados.

La aplicación se inicia realizando un proceso de negociación entre agentes. Este proceso tiene como finalidad seleccionar cuáles nodos de la red reúnen las condiciones para ejecutar las tareas de procesamiento, es decir, que nodos tienen los suficientes recursos de memoria principal y secundaria, procesadores, recursos gráficos, y conexión de la red, para ejecutar la aplicación asignada. En este proceso participa el AAA, y los agentes AGRs que se encuentran dispersos en los otros nodos de la red.

Por su parte, el AGA realiza el trabajo de elección del conjunto de agentes especializados que pueden procesar los datos y devolver los resultados esperados. Uno de los esquemas más utilizados para este tipo de procesamiento son las redes neuronales, pero existen otros esquemas tales como los árboles de decisión y métodos de aproximación que también pueden ser aplicados. Además de poder aplicar diferentes tipos de procesamiento, también es posible utilizar diferentes tipos de implementación, para el ejemplo, se utilizan dos tipos de implementaciones para redes neuronales: una desarrollada en Java y otra basada en un componente COM [11] que se comunica con MATLAB® (una aplicación comercial que provee herramientas para el cálculo matemático)[2]. Cuando se consulta las aplicaciones disponibles, el AGA sugiere estos dos agentes que están vinculados a dos aplicaciones: NNAgent y NNMatlabAgent. Luego del procesamiento, los resultados son devueltos al agente interfaz (UIAgent), y finalmente, mostrados al usuario (ver figura 4).

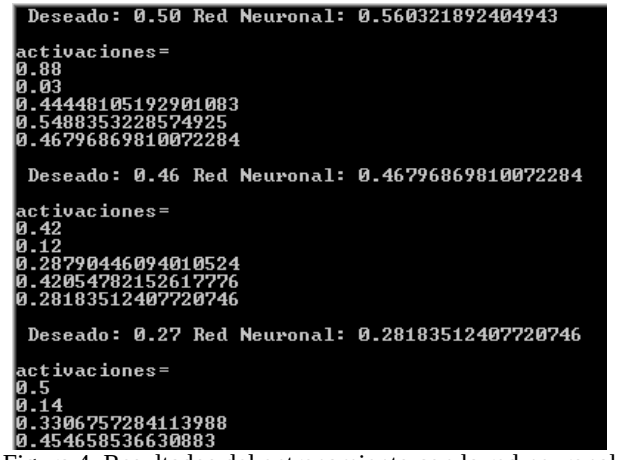


Figura 4. Resultados del entrenamiento con la red neuronal

6. CONCLUSIÓN

Aunque el modelo basado en agentes aporta líneas claras para la construcción de sistemas de software, se hace necesario tomar en cuenta el aspecto integracional. Así, la plataforma de agentes debe contar con un conjunto de servicios, propuesta del MGS, que permitan de manera eficiente la conexión con recursos y aplicaciones ya

[2] Para mayores detalles de esta aplicación puede consultar la dirección web: http://www.mathworks.com

existentes, una capa middleware que resuelva la integración, lo que es una herramienta indispensable para la implantación del paradigma de agentes en las plataformas computacionales actualmente operativas. Particularmente, para el SCDIA es vital el MGS ya que brinda integración entre cualquier plataforma computacional basada en componentes, servicios, u otro tipo de tecnología actualmente utilizada en la industria o empresa, y los agentes que componen al SCDIA.
El medio de gestión de servicios brinda características y funcionalidades en sus diferentes niveles. En el nivel interfaz se encuentran los servicios asociados con los agentes, que involucran las capacidades de comunicación, asociación y autonomía basada en modelos de inteligencia. En el nivel medio se cuenta con los servicios asociados con los sistemas distribuidos, tales como transparencia, seguridad, nombramiento, interoperabilidad, migración y mensajería. En el último nivel, o nivel de acceso a recursos, se cuenta con un modelo que permite integrar recursos heterogéneos, bajo una misma interfaz utilizando el concepto de reflexión, lo que permite un enlace y sincronización en tiempo de ejecución entre agentes y componentes. La ventaja de nuestra propuesta es que puede ser usado por cualquier otro SMA diferente al SCDIA, y puede ser incorporado sobre cualquier Sistema Distribuido.

Reconocimiento

Este trabajo fue financiado por el Proyecto 97003817: *"Esquemas generales basados en Sistemas Inteligentes para Control de Procesos"*, del programa Agenda Petróleo del CONICIT, y por el Proyecto I-620-98-02-AA: *"Gestión de Recursos en Redes de Estaciones de Trabajo mediante Sistemas MultiAgentes"* del CDCHT de la Universidad de los Andes.

Referencias

[1] Aguilar J., Cerrada, M. et al. Aplicación de Sistemas Multiagentes en Problemas del Mundo Real. *XXVII Conferencia Latinoamericana de Informática (CLEI).* Merida, Venezuela, 2001.

[2] Aguilar J., Cerrada, M. et al. Application of the Agent Reference Model for Intelligent Distributed Control Systems. in: *Advances in Systems Science: Measurement, Circuits and Control.* Edited by: N. Mastorakis and L.A. Pecorelli. World Scientific and Engineering Society Press, pp. 204-210, 2001.

[3] Bigus J., Jennifer B. Constructing Intelligent Agents using Java. *Wiley Computer Publishing*. 2001.

[4] Bellifemine F., Poggi A., Rimassi G. JADE: A FIPA-Compliant agent framework, *Proceeding Practical Applications of Intelligent Agents and Multi-Agents,* 1999. (http://sharon.cselt.it/projects/jade)

[5] Hagan M, Demut H., Beale M. Neural Network Design. *Publishing Company*. 1996.

[6] Iglesias C. Definición de una metodología para el desarrollo de sistemas multiagentes. *Tesis Doctoral.* Universidad Politécnica de Madrid. Enero 1998.

[7] Nillson, N. Inteligencia Artificial: Una nueva síntesis. *McGraw Hill.* 1998. España.

[8] Rivas, F., Aguilar, J. et al. Especificación Detallada de los Agentes del SCDIA, *Tercer Informe Técnico, Proyecto Agenda Petróleo No. 97003817,* Universidad de los Andes. 2003.

[9] Rusell S., Norvig P. Inteligencia Artificial: Un enfoque moderno. *Prentice Hall*. 1995. México.

[10] Tanenbaum A. Sistemas Operativos Distribuidos. *Prentice Hall.* 1ra Edición. 1995. México.

[11] Templeman J. Beginning MFC COM Programming. *Wrox Press.* 1997.

[12] Weiss G. Multiagent Systems. *MIT Press.* 1999.

Descripción de un Sistema Manejador de Objetos Web

José L. Aguilar,
Centro de Microcomputación y Sistemas Distribuidos (CEMSID)
Escuela de Ingeniería de Sistemas, Facultad de Ingeniería
Universidad de los Andes
Mérida – Venezuela
aguilar@ula.ve

y

Juan Vizcarrondo
Postgrado de Computación. Facultad de Ingeniería
Universidad de los Andes
Mérida – Venezuela.

Abstract

The number of applications, systems and services developed for the Web is very large. In some cases, there are not supports at the level of the operating systems for them. One alternative is to develop a model of operating system, called SOW, which supports and manages a set of services in a heterogeneous and dynamic environment like Internet. One of the subsystems of this operating system must be the Management System of Web Object. It manages the web objects migration and the web objects replication on the SOW. In this paper we present the design of this subsystem.
Keywords: Operating Systems, Web Objects Migration, Web Objects Replication, Management System of Web Object.

Resumen

La cantidad de sistemas, servicios y aplicaciones desarrolladas para la web han crecido considerablemente. En algunos casos, el soporte por parte de los sistemas operativos existentes a cada uno de ellos no es el esperado. Como alternativa de solución a esta necesidad, se plantea un modelo del sistema operativo denominado SOW, el cual soporta y maneja un conjunto de servicios en un contexto heterogéneo, dinámico y adaptativo, bajo el enfoque de reconfiguración de las aplicaciones. El SOW esta conformado por cuatro subsistemas que llevan a cabo una serie de funciones coordinadas. Uno de estos subsistemas es el Subsistema Manejador de Objetos Web, el cual es el encargado de la migración y replicación de los objetos web existentes en la SOW. En este trabajo presentaremos dicho subsistema.
Palabras claves: Sistemas Operativos Web, Migración de Objetos, Replicación de Objetos, Manejo de Objetos Web.

1. INTRODUCCIÓN

Dada la amplia variedad de servicios inimaginables que surgen cada día en la web, resulta difícil diseñar un sistema operativo que apoye y use a cada uno de esos servicios. De éstas necesidades surge un área llamada Sistema Operativo Web (SOW), que tiene como objetivo principal proveer una plataforma que permita a los usuarios beneficiarse del potencial computacional ofrecido en la web, a través del compartimiento de recursos y de la resolución de los problemas de heterogeneidad y adaptabilidad dinámica presentes en la misma [1, 6, 9,11, 13, 14]. Así, para alcanzar un rendimiento óptimo en un ambiente dinámico de recursos distribuidos como la Internet, el SOW debe ser configurable y capaz de adaptarse a los cambios en cuanto a la disponibilidad de recursos (de software y de hardware). Teniendo en cuenta esas consideraciones, el modelo de SOW presentado en [1] propone una serie de aspectos para proveer servicios que se adecuen a esos rasgos especiales de la web. Así, nuestro SOW presenta un diseño que cuenta con las herramientas asociadas para permitir el uso transparente e interactivo de los recursos accesibles a través de la red, en cualquier momento que un usuario lo requiera. Esos servicios pueden ser hardware, software, o una combinación de ambos. El usuario sólo necesita comprender la interfaz del SOW, sin importarle como su solicitud es satisfecha.

Existen varias propuestas para el manejo e integración de los recursos computacionales disponibles en la Web. Quizás el proyecto más general sea el ***WOS^{TM}*** [11], ya que permite el manejo e integración de los recursos tratando el problema de la heterogeneidad y volatilidad en la Web. Este proyecto, al igual que nuestra propuesta de SOW, está basado en la idea del uso de versiones como solución a esos problemas. En [11] se ha planteado la creación de una interfase llamada WOSMPI, que le permitirá al ***WOS^{TM}*** migrar objetos en Internet, para que puedan ser usados por aplicaciones de calculo científico, donde se requiere el movimiento de enormes cantidades de datos. Otra propuesta para el manejo de recursos en la Web parecida, es la arquitectura ***Jini*** de SUN Microsystems [12]. ***Jini*** provee servicios de localización de recursos (conocidos como *lookups*), y aplica la idea de agrupar recursos en federaciones.

El objetivo del Subsistema Manejador de Objetos Web (SMOW) es proveer al SOW de mecanismos que le permiten adaptarse a la naturaleza dinámica de los recursos en Internet. Para esto, nuestro SOW se adapta a la volatilidad de los recursos e intenta disminuir la latencia debido a la distancia entre los nodos, tratando de tener más cerca de los usuarios la información que requieren, valiéndose de mecanismos como la replicación y migración de objetos. Así, la *replicación de objetos* es usada para colocar cerca de los usuarios la información que requieren valiéndose de caches en la web. Las caches en la Web tienen la capacidad de almacenar objetos cerca de los usuarios, para que no sea necesario desplazar un objeto de su ubicación original cada vez que se requiera [2, 3, 15]. Por otro lado, la *migración de objetos* provee un mecanismo que permite mejorar la disponibilidad de los recursos, al equilibrar la carga de trabajo entre los diferentes nodos migrando los objetos hacia nodos ociosos, y al reducir las comunicaciones entre los diferentes nodos al colocar los objetos en los sitios de mayor demanda [4, 7, 17]. En Internet existen dos tipos de objetos [5]: objetos estáticos que son colocados en un servidor (fotos, textos, archivos HTML, etc.) y objetos dinámicos que son construidos por programas que se ejecutan cuando son llamados. La migración de los objetos estáticos resulta fácil de implementar, ya que solo es mover un archivo a un nodo [5]. La movilidad de objetos dinámicos por Internet puede realizarse de 2 formas [5]: la primera se refiere a la migración de pequeños fragmentos de código del objeto (procedimientos) a sitios remotos que lo requieran, la segunda describe la migración total del código del objeto a otros sitios.

Finalmente, existen varias técnicas para localizar objetos después de que han sido migrados. La primera se refiere a enviar un mensaje a todos los nodos de la red anunciando la nueva dirección cada vez que se realiza una migración, la otra se refiere a enviar un mensaje global de búsqueda cada vez que se encuentre una referencia no válida [4, 7, 17], pero estas dos técnicas son muy costosa en sistemas de gran escala como Internet, al enviar un mensaje a cada nodo del sistema. Para solventar esta situación existe una técnica denominada Lazos de Persecución [4, 7, 17], la cual consiste en que cuando un objeto migra se deja en su sitio de origen una indicación de la nueva localidad, facilitando así la localización del objeto. Cuando llega un mensaje solicitando el objeto este es redireccionado a su nueva posición. El problema de esta técnica es en sistemas muy grandes donde un lazo puede ser muy largo, o no ser consultado nuevamente debido a que todos los que tienen vínculos con este objeto ya actualizaron su nueva dirección (o por lo menos parte del lazo). En este articulo nosotros proponemos un mecanismos para resolver este problema que llamamos *trazas de objetos*.

Los SMA han representado una solución para el diseño de sistemas muy grandes donde no se puede realizar un control centralizado y sus ambientes son heterogéneos, abiertos y cambian dinámicamente, como Internet [16]. Por estas razones se propone realizar el Diseño del SOW basado en la teoría de agentes. En este articulo presentamos nuestra propuesta de SMOW usando SMA. Comenzamos realizando una breve introducción del SOW. En la siguiente sección presentamos el diseño del SOW usando SMA. Finalmente, en la ultima sección definimos los agentes que componen el SMOW.

2. ARQUITECTURA DEL SISTEMA OPERATIVO WEB (SOW)

El SOW propuesto posee las siguientes características [1]:

- *Distribuido y Versionado:* Diferentes Aplicaciones Distribuidas y varias versiones de los servicios están corriendo simultáneamente sobre los diferentes sitios de la red.

- *Dinámico:* La web es una entidad que está evolucionando permanentemente, por lo tanto, el SOW debe adaptarse a los cambios imprevistos. Es por ello que nuestro SOW tiene incorporado cierto dinamismo en los subsistemas que lo integran. Por ejemplo, a través de la configuración dinámica de los servicios que ofrece; a través de la actualización permanente de la información sobre los recursos locales y remotos disponibles en cada nodo; a través de la agrupación dinámica de los nodos existentes de acuerdo con ciertas características, y a través de la migración, replicación y seguimiento de objetos web.
- *Abierto:* Nuestro SOW se caracteriza como un sistema abierto desde dos puntos de vista. En primer lugar, aceptará que diversas tecnologías sean usadas en todos los niveles de la red (heterogeneidad). En segundo lugar, permitirá que cualquier nodo de Internet pueda ser incorporado al sistema.
- *Inteligente:* Cada uno de los subsistemas del SOW tendrá algún nivel de inteligencia para el desarrollo de alguna de sus funciones con el fin de optimizar su funcionamiento.

Así, nuestro SOW es un sistema operativo versionado e inteligente que se autoconfigura dinámicamente para permitir un acceso fácil y transparente a los recursos distribuidos sobre la Internet. Nuestro SOW utilizara un Middleware orientado a objetos que debe proveer la estructura necesaria para lograr la descripción de todos los recursos en Internet. Nuestra arquitectura de SOW propuesta en [1] esta compuesto por un conjunto de 5 subsistemas:

- *Subsistema Manejador de Recursos (SMR):* Este subsistema maneja todos los requerimientos hechos al SOW, este funciona como un sistema reactivo de manejo de demandas a través del cual se administran los recursos de un ambiente computacional heterogéneo como la Web.
- *Subsistema Manejador de Repositorios:* En el subsistema es donde se depositan todos los recursos con los que cuenta un nodo, este a su vez se divide en dos subsistemas:
  a) Subsistema Manejador de Repositorios Local (SMRL): Funciona como un catalogo de los recursos y versiones del dominio local
  b) Subsistema Manejador de Repositorios Remotos (SMRR): En este subsistema se almacena información sobre los recursos remotos mas utilizados por el sistema local, así como los objetos replicados.
- *Subsistema Manejador de Objetos Web (SMOW):* Es el encargado de realizar la migración y la replicación de objetos en el SOW, para aumentar la disponibilidad del sistema al situarlos en los sitios de mayor demanda o equilibrar la carga. Así, el SMOW debe realizar la selección de los objetos a migrar o replicar, del nodo destino donde se almacenaran, y debe tener implementado los mecanismos necesarios que permitan localizar los objetos migrados, y que garantice la migración y replicación de los objetos.
- *Subsistema Manejador de Comunidades (SMC):* Comisionado para agrupar los diferentes nodos que componen el SOW en grupos que presenten las mismas características.

3. PROPUESTA DE NUESTRO SOW COMO SISTEMA MULTIAGENTES

El diseñar un SOW conlleva tratar con el problema de heterogeneidad presentes en los múltiples servicios que se ofrecen en la web, tanto a nivel de hardware como de software presentes en los diferentes sitios, y con el problema de adaptabilidad debido a la volatilidad de los recursos. Diseñar un sistema de control centralizado para solucionar estos problemas resulta imposible. Los SMA han representado una solución para el diseño de sistemas muy grandes donde no se puede realizar un control centralizado y sus ambientes son heterogéneos, abiertos y cambian dinámicamente como Internet [16]. Por estas razones se propone realizar el diseño de nuestra propuesta de SOW usando SMA. Los SMA proveen métodos dinámicos y mecanismos de inteligencia, que permiten el diseño del SOW como un sistema robusto, donde todos sus componentes asumen sus propios roles y cooperan entre sí para la solución de problemas.

Realizar el diseño del SOW basado en SMA facilita el desarrollo de cada uno de los subsistemas del SOW por separado, logrando reducir la complejidad requerida para solucionar el problema de gestión a nivel global. Así, al diseñar nuestro SOW como un SMA, visto desde el nivel mas alto el SOW puede ser considerado como un solo componente, denotado por L_0, donde la información interna y los procesos en él no son especificados (se ocultan las comunicaciones y los procesos internos del SMA). En el próximo nivel de abstracción L_1, el componente L_0 puede ser visto como una colección de subsistemas que cumplen con los objetivos del SOW (ver figura 1).

Sistema Multiagentes (L0)	Esta compuesto por (L1)	Satisface Objetivos	Interrelacionado con
SOW	SMR	Administrar e integrar recursos computacionales presentes en el SOW	Usuarios, SMRL, SMRR y SMC
	SMRL	Buscar información en los RL	SMR, SMRR y SMOW
		Mantener coherente la información existente en los RL Y rr	
		Administrar el uso eficiente de los recursos en los RL y RR	
	SMRR	Buscar Información en los RR	SMR, SMRL, SMOW
		Mantener actualizada y coherente la información en los RL	
	SMOW	Replicar objetos en los RR	SMRL, SMRR y SMC
		Migrar objetos hacia nodos de mayor demanda	
	SMC	Agrupar nodos con afinidades en comunidades	SMR, SMOW

Figura 1. Descomposición del SOW en 5 SMA

4. PROPUESTA DEL SUBSISTEMA MANEJADOR DE OBJETOS WEB (SMOW)

En nuestra propuesta de SOW, las replicas son almacenadas en los RR y los objetos web migrados en los RL. El rol desempeñado por el SMOW en lo que respecta a replicación, es el de proveer los mecanismos necesarios para transportar las replicas en Internet hacia los RR, cada vez que es solicitada por el SMRR, además de mover la información necesaria hacia los RL, cada vez que sea requerida por el SMRL, para garantizar la coherencia de los objetos cuando un objeto ha pasado a ser incoherente por la modificación de una replica.

A nivel de la migración de objetos, este es el acto de transferir un objeto entre dos máquinas (entre el nodo origen y el nodo destino), buscando maximizar algún objetivo, como por ejemplo, el balance de la carga entre los nodos. Para esto es necesario proveer un mecanismo que defina que objeto migrar y hacia donde. En nuestra propuesta es necesaria la migración de objetos para aumentar la disponibilidad de los objetos del sistema, además de reducir las comunicaciones entre los diferentes nodos. El SMOW es el encargado de realizar la selección de los objetos y el nodo destino donde se almacenaran, además de la implementación del mecanismo que garantice el transporte de los objetos por los distintos nodos del SOW. Cuando se realiza la migración de objetos, se debe garantizar la localización de estos cada vez que un objeto abandona un nodo y se almacena en otro, para esto en nuestra propuesta de SOW se ha propuesto una estructura basada en trazas, las cuales van desapareciendo a medida que transcurre el tiempo.

Las características a tomar en cuenta en el diseño del SMOW del SOW son:

- Cada Comunidad de nodos del SOW posee un SMOW que administra el movimiento de los objetos y de las replicas en toda la comunidad.
- El SMOW provee los mecanismos para la creación de replicas de objetos y suministrárselos a todos los RR que los soliciten.
- El SMOW provee mecanismos que le permiten realizar la migración de los objetos web en todo el SOW, el cual realiza por petición del SMC. Además, es el encargado de la creación de trazas de objetos en los RL cada vez que se produce la migración de objetos.

A continuación se definen las bases fundamentales de nuestra propuesta de diseño del SMOW, precisando los mecanismos necesarios para realizar la replicación, migración y facilitar la localización de objetos cada vez que son migrados.

4.1. Replicación de Objetos

En nuestro SOW, el SMOW recibe peticiones de replicar un objeto cada vez que algún SMRR necesite almacenar una nueva replica en el RR. El encargado de la selección de los objetos que se van a replicar en el RR es el SMRR, para esto, cada vez que el SMRR necesita hacer una nueva replica se la solicita al SMOW. Cada vez que el SMRR solicita que le sea suministrada la replica, el SMOW revisa el objeto en el RL para verificar que no se encuentra en uso en ese momento, lo bloquea, y transfiere una copia de su contenido al RR. En caso que el objeto se encuentre en uso, el SMOW responde al SMRR que intente más tarde la solicitud.

4.2. Migración de Objetos

Para poder realizar la migración de un objeto es necesario conocer previamente la selección del objeto a ser migrado, el sitio de destino, y el mecanismo que permita la localización del objeto [10, 17]. En nuestra propuesta los objetos son migrados para colocarlos en los sitios de mayor demanda, migrando aquellos objetos web dinámicos. Para esto, el SMC cada vez que requiere la migración de un objeto lo solicita al SMOW, el SMOW realiza una comparación entre la demanda en el SMRL origen y el posible destino, y si es mayor la demanda en el SMRL destino se migra el objeto hacia ese nodo. Existen muchos mecanismos para realizar la migración de los objetos [4, 7, 17], en este trabajo se ha seleccionado el mecanismo de la *migración impaciente*, ya que el objeto y las comunicaciones recibidas son transferidas completamente hacia el nodo destino para de esta forma no hacer necesaria que la comunicación con el nodo origen sea restablecida nuevamente, al contrario de lo que ocurre con la migración perezosa y de la copia previa, las cuales solo transportan inicialmente una parte del objeto hacia el nodo destino.

4.3. Localización de Objetos Migrados

Existen varias técnicas para localizar objetos después de que han sido migrados [10, 17]. En este trabajo se propone un nuevo mecanismo basado en *trazas de objetos*, inspirado en los Sistemas de Colonias de Hormigas, los cuales son modelos que emulan el comportamiento de colonias de hormigas reales [8]. Las hormigas son capaces de seguir la ruta más corta en su camino de ida y vuelta entre la colonia y una fuente de alimento. Esto es debido a que las hormigas pueden "transmitirse información" entre ellas gracias a que cada una de ellas, al desplazarse, va dejando un rastro de una sustancia llamada feromona (traza) a lo largo del camino seguido, la cual es detectada por otras hormigas, tendiendo a seguir la traza con mayor cantidad de feromona. A su vez, éstas van dejando su propia feromona a lo largo del camino recorrido, y por tanto, lo hacen más atractivo puesto que se ha reforzado el rastro de feromona. Por otro lado, la feromona también se va evaporando con el paso del tiempo provocando que el rastro de feromona sufra cierto debilitamiento. La noción de trazas de feromona dejada por los objetos, es la base de nuestra propuesta de localización de los objetos a implementar en el SOW. El objeto al moverse va dejando el rastro de sus movimientos, lo que podría utilizarse para encontrarlo al ser requerido por un RR. Pero por otro lado, el rastro dejado por los objetos al desplazarse debe ir desapareciendo a medida que transcurre el tiempo.

Nuestra propuesta es que cada vez que un objeto es migrado de algún SMRL de nuestro SOW, es colocado en el RL un lazo de persecución, que permita a los objetos encontrarlos en su ubicación actual. El problema de implementar los lazos de persecución es que a medida que transcurre el tiempo, se van convirtiendo en información no coherente o no son utilizados [17], ya que los objetos a que hacen referencia han migrado a otros sitios o no son utilizados por las caches ya que han actualizado el apuntador al objeto. Para solventar esta situación es que nosotros usamos la noción de las trazas de las hormigas. Los lazos de persecución dejados en el SMRL, consisten en una estructura que contiene la intensidad de la feromona, la cual se va evaporando a medida que transcurre el tiempo hasta desaparecer. Cuando un nodo accede a este lazo de persecución para actualizar la ubicación del objeto, la cantidad de feromona es reforzada (aumentada) por el SMRL, ya que si un SMRR ha actualizado la ubicación en un momento, es posible que otros SMRR aun no lo hayan hecho. Si este lazo no es consultado nuevamente, la cantidad de feromona que contiene el lazo se va evaporando, hasta desaparecer, eliminando de esta manera los lazos basura que no van a ser consultados nuevamente. En nuestro SOW propuesto, el encargado de crear las trazas es el SMOW, y los encargados de actualizarlas son tareas tanto del SMRL como del SMRR.

4.4. Roles Desempeñados por el SMOW en el SOW

Los roles en los que se encuentran involucrado el subsistema SMOW del SOW se muestran en la Figura 2.

Acción	Decide	Solicita	Implementa	Almacena	Busca	Actualiza
Migración	SMOW	SMC	SMOW	SMRL destino	-	-
Replicación	SMRR	SMRR	SMOW	SMRR	-	SMRR y SMRL
Trazas	SMOW	SMOW	SMRR y SMRL	SMRL origen	SMRR	SMRR y SMRL

Figura 2. Roles Desempeñados por el SMOW

La migración en el SOW permite ubicar los objetos web en los sitios donde más se les demanda en Internet, para esto el SMOW recibe las peticiones de migración de objetos del SMC y luego realiza la inferencia de si es necesario migrar el objeto al nuevo destino o no. En caso de decidir la migración, el SMOW la implementa almacenando el objeto en el SMRL destino y eliminándolo del SMRL origen. Una vez que el objeto ha sido migrado, la intensidad de las trazas son almacenadas en el SMRL origen, por solicitud del SMOW. La actualización de la traza es responsabilidad del SMRL al recibir la solicitud de acceso al objeto por parte del SMRR, o por el proceso de evaporación de ella.

La replicación de objetos le permite al SOW manejar la disponibilidad de los recursos, además de disminuir el tráfico en la red. Las replicas son almacenadas en los SMRR del SOW, por tanto, estos son los encargados de realizar la solicitud de replicas en el SOW. Además, debe administrar el espacio y la coherencia de las replicas que se encuentran en los RR. El SMOW recibe la solicitud de replicación por parte del SMRR y la implementa replicando el Objeto Web solicitado en el RR. Además, la asignación y desasignación de replicas es tarea del SMRR, la cual realiza por solicitud del SMR.

5. DESCRIPCIÓN DE LOS AGENTES DEL SMOW USANDO SMA

Realizar el diseño del SOW basado en SMA facilita el desarrollo de cada uno de los subsistemas del SOW por separado, logrando reducir la complejidad requerida para solucionar el problema de gestión a nivel global. Particularmente, la identificación y descripción de los agentes que intervienen en el SMOW se realiza en base a los actores y casos de usos definidos para el SMOW. Además, es necesario definir las estructuras de datos que usara el SMOW. A continuación se presenta la arquitectura propuesta para el SMOW.

5.1. Arquitectura del SMOW

En el SMOW del SOW las operaciones son coordinadas por dos componentes fundamentales (Figura 3), éstos son: el Replicador de Objetos y el Migrador de Objetos. Estos dos componentes constituyen las unidades funcionales básicas de la arquitectura del SMOW.

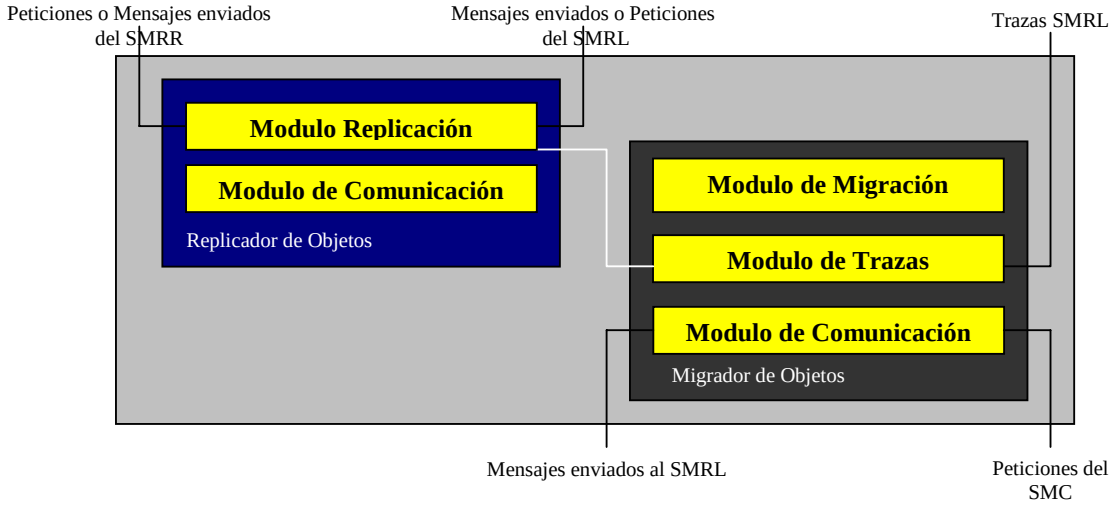


Figura 3. Componentes del SMOW

A continuación se presenta la descripción de cada uno de estos componentes:

- *Replicador de Objetos:* Esta unidad tiene como objetivo el traslado de las replicas de los objetos en el SOW. Para esto utiliza dos módulos, e*l Modulo de Replicación*, que le permite realizar las replicas; y el *Modulo de Comunicación*, que le permite la coordinación con el SMRL fuente y el SMRR destino.
- *Administrador de Traslado de Objetos:* Esta unidad tiene como objetivo la migración de objetos web en el SOW y la creación de trazas. Para esto utiliza tres módulos, el *Modulo de Migración de Objetos*, que le permite realizar el traslado de Objetos de un SMRL a otro SMRL, el *Modulo de Trazas*, que le permite la creación de trazas cuando un objeto es migrado; y el *Modulo de Comunicación*, que le permite la interacción con el SMRL fuente en el momento de iniciar la migración.

5.2. Estructura de Datos Utilizada por el SMOW

Para facilitar el manejo de los objetos Web es necesario la creación de estructuras de datos. Para esto, el SMOW posee 2 estructuras de datos que le permiten manejar los objetos en el RL y RR. La figura 4 se refiere a la estructura de datos utilizada por el SMOW para manejar los objetos en el RL, la cual le concede el poder de bloquear y cuantificar la demanda de los objetos en el RL. La estructura de datos de la figura 5 le permite al SMOW reiniciar los objetos en el RR una vez que han sido migrados.

Recurso	Versión	Frecuencia	Estado

Figura 4. Estructura de datos utilizada para el manejo de objetos en el RL.

El significado de los campos que conforman la estructura de datos que utiliza el SMOW para el manejo de los objetos Web en el RL es el siguiente:

Frecuencia: Se refiere a la cantidad de veces que es requerido un objeto Web en el RR. Es un vector que almacena cantidad de demandas y origen de la misma.
Recurso: Identificador del objeto Web.
Estado: Se refiere al estado actual del objeto Web, lo cual es usado para la coherencia de estos. Dichos estados pueden ser:

- Modificado.
- Invalido.
- Exclusivo.
- Compartido.

El **Estado** del objeto es usado para bloquear el objeto en el RL, para que no pueda ser modificado por el SMRL, mientras se realizan operaciones de replicación o migración sobre él. El campo **Recurso** facilita la identificación de los objetos en el RL. La **Frecuencia** del objeto se refiere a la demanda del objeto en el RL, el cual le permite al SMOW tomar decisiones de migración sobre los objetos del RL, particularmente sobre si hay que hacerla y hacia donde.

Frecuencia	Recurso	Estado	Versión	Ubicación

Figura 5. Estructura de datos utilizada para el manejo de las replicas en el RR.

El significado de los campos que conforman la estructura de datos que utiliza el SMRR para el manejo de las replicas en el RR es el siguiente:
Frecuencia: Se refiere a la cantidad de veces que es requerido un objeto Web en el RR desde el sitio local.
Recurso: Identificador del objeto Web.
Estado: Se refiere al estado actual del objeto Web, lo cual es usado para su coherencia. Dichos estados pueden ser:

- Modificado.
- Invalido.
- Exclusivo.
- Compartido.

Versión: Versión del objeto web.
Ubicación: Sitio Remoto en donde se encuentra localizado el objeto replicado.

La **frecuencia** permite cuantificar la demanda de una determinada replica, la cual es usada por el SMRR para la toma de decisiones de cuales replicas reemplazar y cuales replicas mantener en el RR. Los campos **Recurso** y **Versión** facilitan la identificación de las replicas en el RR. Por ultimo, los campos **Estado** y **Ubicación** son utilizados por el SMRR para el manejo de coherencia de las replicas, y por el SMOW para reiniciar la nueva replica en el RR. El **Estado** de un objeto se refiere al estado actual en que se encuentra la replica en ese momento y la **Ubicación** almacena el apuntador del nodo donde se encuentra ubicado el objeto replicado.

El almacenamiento de las trazas de los objetos Web en el RL se realiza en el formato que se muestra en la Figura 6.

Recurso	Versión	Tasa de evaporación	Intensidad Traza	Ubicación

Figura 6. Estructura de datos utilizada para almacenar las trazas de los objetos en el RL.

El significado de los campos que conforman la estructura de datos que utiliza el SMOW para almacenar las trazas de los objetos Web es el siguiente:
Recurso: Identificador del objeto Web.
Versión: Versión del objeto Web.
Tasa de Evaporación: La tasa de evaporación que permite eliminar la traza en el transcurso del tiempo.
Intensidad Traza: La cantidad de feromona del objeto.
Ubicación: Ubicación en donde se encuentra el objeto.

Los campos **Recurso**, **Versión** y **Ubicación** tienen el mismo significado de los campos de las estructura de datos utilizadas para el manejo de los objetos y las replicas en el RL y RR. El campo **Intensidad Traza** contiene la intensidad de la feromona que se encuentra en el lazo de persecución, y la **tasa de Evaporación** contiene la velocidad con que se evapora el lazo.

5.3. Descripción de los Actores y Casos de Uso

Los actores son descritos por el papel que desempeñan en el sistema, identificando posteriormente los agentes que en él se desenvuelve y las colaboraciones que realizan entre ellos. En el subsistemas SMOW es posible la descripción de dos actores, los cuales se muestran en la Figura 7.

Actor	Descripción	Casos de Uso
Replicador de Objetos	Provee mecanismos que le permiten realizar replicas.	- Replicar Objeto
Migración de Objetos	Provee mecanismos que le permiten realizar la migración de objetos.	- Migrar objetos - Crear Traza

Figura 7. Descripción de los actores que intervienen en los subsistemas SMOW

A continuación se muestra la descripción de los casos de uso de los actores anteriormente mencionados:

5.3.1. Casos de uso del Actor Replicación de Objetos

Los casos de uso del actor Replicador de Objetos se muestran en la Figura 8.

Caso de Uso: Replica de Objeto. **Resumen:** El actor Replicador de Objetos recibe la solicitud de la creación de una replica por parte de un SMRR, y luego realiza la replica del objeto para almacenarla en el SMRR del nodo solicitante. **Actores:** SMRR, SMRL, Replicador de Objetos. **Precondición:** Haber recibido una solicitud de Replica. **Excepción:** No se puede realizar la replica del objeto.

Figura 8. Casos de uso del actor Replicador de Objetos.

5.3.2. Casos de uso del Actor de Migración de Objetos

Los casos de uso del actor Migración de Objetos se muestran en la Figura 9.

Caso de Uso: Migración de Objeto. **Resumen:** El actor de Migración de Objetos realiza la migración de objetos web, la cual efectúa hacia los sitios de mayor demanda. **Actores:** Migración de Objetos, SMC y SMRL. **Precondición:** Haber recibido una solicitud de migración. **Excepción:** No se puede realizar la migración del objeto.	**Caso de Uso:** Crear Traza. **Resumen:** El actor de Migración de Objetos solicita al SMRL origen del objeto la creación de la traza de objeto cuando se realiza la migración de un objeto. **Actores:** SMRL, Migración de Objetos. **Precondición:** Objeto migrado. **Excepción:** No se puede crear la traza en el SMRL origen.

Figura 9. Casos de uso del actor de Migración de Objetos.

5.4. Identificación de los Agentes del SMOW

La identificación y descripción de los agentes que intervienen en el sistema SMOW se realiza en base a los actores y casos de usos definidos. Según los roles definidos hasta esta fase, tenemos que el SMOW provee las siguientes funcionalidades: Replicar Objetos, Migrar Objetos y Crear Traza. Estas funciones corresponden a los roles de cada uno de los actores identificados. El SMA aquí propuesto para el SMOW consta de dos agentes, los cuales corresponden a los actores identificados anteriormente: Agente Replicador de Objetos (ARO) y Agente Migrador de Objetos (AMO). El agente ARO contendrá mecanismos que le permitirán realizar las replicas, la cual realiza por petición del SMRR. El agente AMO proveerá mecanismos que le permitirán inferir los sitios de mayor demanda para un objeto, además de ser el encargado del traslado de los objeto hacia dichos sitios. Cuando un objeto es migrado, el agente AMO crea una traza en el SMRL donde se encontraba el objeto.

Las tareas que ejecutara el SMOW se derivan de los actores y casos de uso realizados por el subsistema en el SOW. Así, las tareas identificadas se muestran en la Figura 10.

Agente	Tareas	Descripción
Administrador de replicación de Objetos (ARO)	Replicar Objetos	El agente ARO recibe la solicitud de replicar un objeto por parte de SMRR, para esto crea una replica del objeto en el SMRL, y luego la traslada al SMRR solicitante.
Administrador de Migración de Objetos AMO	Migrar Objetos	El agente AMO recibe peticiones de mover un objeto hacia otro nodo, este decide si migrar el objeto hacia el destino sugerido en base a las demandas de los dos nodos y si el nodo origen presenta problemas. Si el agente AMO decide migrar el objeto, este mueve el objeto hacia el SMRL destino.
	Crear Traza	El agente AMO necesita crear trazas cada vez que realiza la migración de un objeto.

Figura 10. Identificación de las tareas del SMOW

5.5. Interacciones de los agentes del SMOW con el resto de los subsistemas del SOW

Los agentes propuestos para el SMOW deben mantener las mismas interacciones propuestas para el subsistema. Así, los agentes del SMOW deben interactuar con los demás subsistemas del SMOW para alcanzar sus objetivos de replicación y migración de objetos. El encargado de realizar la replicación de objetos es el agente ARO, para esto, recibe la solicitud de creación de una replica por parte del SMRR, luego bloquea el objeto, cambiandole su estado para que el SMRL origen no pueda realizar operaciones sobre él, para después transportarlo hacia el RR solicitante. Las interacciones de los actos de habla de la conversación Replicar Objeto en el RR se presentan en el diagrama UML de la figura 11.

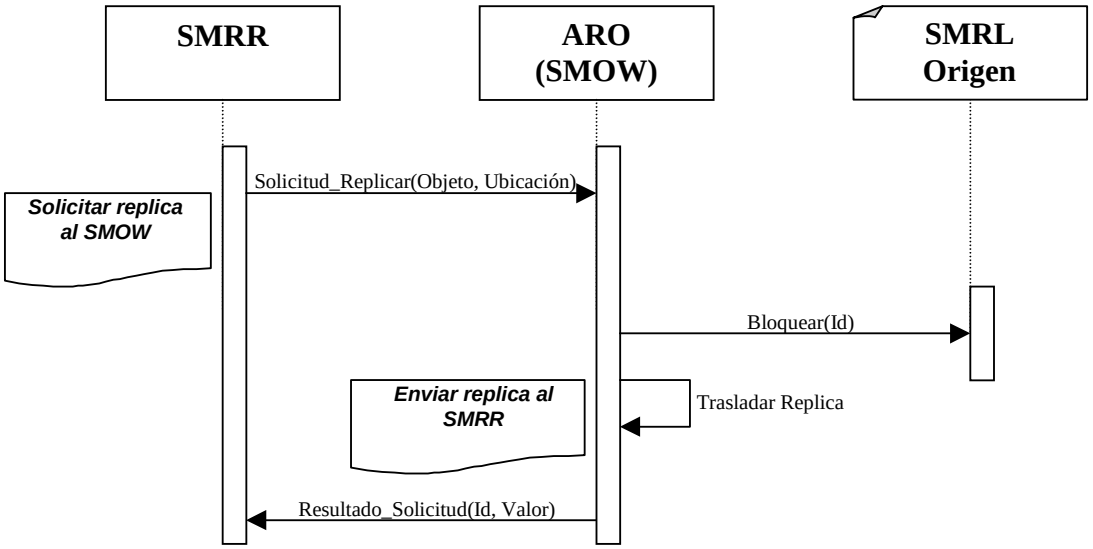


Figura 11. Diagrama UML de la conversación Replicar Objeto.

El agente AMO es el encargado de colocar los objetos presentes en el SOW en los sitios de mayor demanda. Así, el agente AMO recibe la solicitud de migración de un objeto por parte del SMC, solicita permiso para migrar el objeto al destino, bloquea el objeto en el RL origen, y lo migra al SMRL destino, después solicita al Replicador de Objetos que cree la traza del objeto en el SMRL origen. Las interacciones de los actos de habla de la conversación Búsqueda de Objeto en el RR se presenta en el diagrama UML en la figura 12.

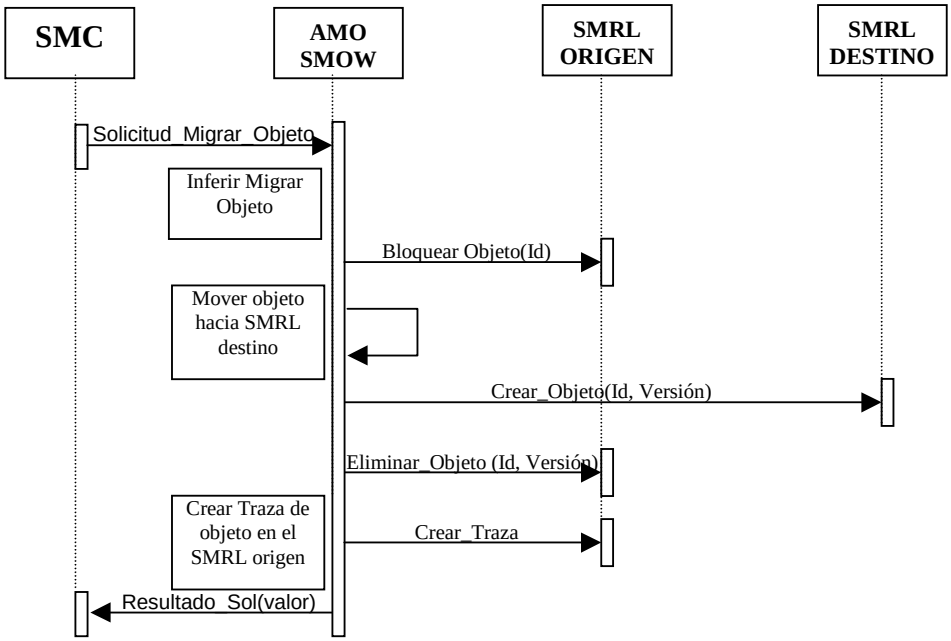


Figura 12. Diagrama UML de la conversación Replicar Objeto.

6. CONCLUSIONES

El SMOW provee al SOW de mecanismos que le permiten adaptarse a la naturaleza dinámica de los recursos en Internet. De esta manera el SOW se adecua a la volatilidad de los recursos e intenta disminuir la latencia debido a la distancia entre los nodos, tratando de tener más cerca de los usuarios la información que requieren, valiéndose de mecanismos como la replicación y migración de objetos.

El realizar el diseño del SOW basado en SMA permite el desarrollo de los subsistemas del SOW por separado, donde todos sus componentes asumen sus propios roles y cooperan entre sí para la solución de problemas, distribuyendo de esta manera, los objetivos del SOW entre los subsistemas que lo componen. Así, el diseño del SMOW fue desarrollado como un SMA separado del resto de los subsistemas, pero donde los agentes obtenidos cooperan e interactúan entre sí y con el resto de los subsistemas del SOW para alcanzar sus objetivos.

El SMOW propuesto consiste en un SMA compuesto por 2 agentes: el ARO y el AMO, los cuales cumplen con los roles y objetivos planteados para el SMOW, como son el permitir la migración y replicación de objetos Web. El agente ARO permitirá al SOW almacenar replicas objetos en los RR. Por otra parte el agente AMO proveerá al SOW de mecanismos que le permitirán migrar objetos hacia los nodos de mayor demanda del SOW, realizando la inferencia necesaria para selección del objeto a migrar y del nodo destino ha donde será enviado. Además, provee mecanismos que le permiten al SOW la localización de los objetos web una vez migrados. Para esto, en este articulo se propuso un mecanismo que permite la localización de objetos, que nosotros hemos llamado trazas de objetos, el cual utiliza la noción de trazas de feromonas usadas por las hormigas en su búsqueda de comida, para eliminar los lazos de persecución basura presentes en los RL.

Reconocimiento

Este trabajo fue financiado por el Proyecto I-620-98-02-AA: *"Gestión de Recursos en Redes de Estaciones de Trabajo mediante Sistemas MultiAgentes"* del CDCHT de la Universidad de los Andes.

Referencias

[1] Aguilar J, Perozo N., Ferrer E., Vizcarrondo J. Arquitectura de un Sistema Operativo Web, *Revista Gerencia Tecnológica Informática*, N. 2, Vol 1, 13 páginas, Colombia Julio de 2003. http://www.cidlisuis.org/aedo/RGTIN2V1/RGTI_01.pdf, Sistema de Autoría Electrónica de Documentos.

[2] Aguilar J. File Decomposition, Replication and Assignment Problems: Definition and Resolution methods. *International Journal of Engineering Intelligent Systems*, Vol. 8, N. 4, pp. 245-254, 2000.

[3] Aguilar J., Leiss L. A Web Proxy Cache Coherency and Replacement Approach, *Lectures Notes in Computer Science*, Vol 2198, pp 75-94,2002.

[4] Artsy Y., Finkcl R. Designing a Process Migration Facility: The Charlotte Experience. *IEEE Computer*, pp. 47-56, 1989.

[5] Baker S., Bongki M. Distributed cooperative web servers**.** *Computer Networks*, Vol. 31, pp. 1215- 1229, 1999.

[6] Baldeschwieler J., Blumofe R., Brewe E. ATLAS: An Infrastructure for Global Computing*. Proceedings of 7th ACM Special Interest Group on Operating Systems, European Workshop on System Support for Worldwide Applications*, pp.165-172, 1.996.

[7] Douglis F., Ousterhout J. Process migration in the Sprite operating system, *Proceedings of the 7th International Conference on Distributed Computing Systems*, IEEE, Berlin, West Germany, pp. 18-25, 1987.

[8] Dorigo, M. Maniezzo V. Coloni A. The ant system: Optimization by a colony of cooperating agents, *IEEE Trans. Syst. Man, Cybern.*, Vol. 26, pp. 29-41, 1996.

[9] Foster I., Kesselman C. Globus: A Metacomputing Infraestructure Toolkit. *Supercomputer Applications*, Vol. 11, N, 2, pp. 115- 128, 1998.

[10] Kon F., Campbell R., Ballesteros F**.** 2k: A Distributed Operating System For Dinamic Heterogeneous Environments. *Proceedings of 9th IEEE International Symposium on High Performance Distributed Computing*, pp. 201-207, Pittsburgh, USA, 2000.

[11] Kropf P. Overview of the WOS Project. *Proceedings of Advanced Simulation Technologies Conference*, pp. 350-356, 1999.

[12] Morgan, S. Jini to the Rescue. *IEEE Spectrum*, Vol. 4, pp:44-49, 2000.

[13] Vahdat A., Belani E., Eastham P., Yoshikawa C. WebOS: Operating System Services for Wide Area Applications. *Proceedings of 7th IEEE Symposium on High Performance Distributed Systems*, pp. 52- 63, 1998.

[14] Van Steen M., Homburg P., Tanenbaum A. The Architectural Design of GLOBE: A Wide-Area Distributed System, *Technical Report IR-422*, Vrije Universiteit Amsterdam, 1997.

[15] Wang Jin, A survey of Web Caching Schemes for the Internet**,** *ACM Computer Communication Review*, Vol 25, N. 9,pp 36-46, 1999.

[16] Weiss G. Multiagent Systems. *The MIT Press*. Massachusets, USA, 1999.

[17] Wheeler R. Migraction Process. *ACM Computing Surveys*, Vol. 32, No. 3 2000.

Extração de Topic Maps no *Oveia*: Especificação e Processamento

Giovani Rubert Librelotto[*]
Universidade do Minho, Departamento de Informática
Braga, Portugal, 4710-057
grl@di.uminho.pt

and

José Carlos Ramalho
Universidade do Minho, Departamento de Informática
Braga, Portugal, 4710-057
jcr@di.uminho.pt

and

Pedro Rangel Henriques
Universidade do Minho, Departamento de Informática
Braga, Portugal, 4710-057
prh@di.uminho.pt

Resumo

Este artigo apresenta uma solução para conquistar interoperabilidade entre sistemas de informação heterogêneos, com o uso de ontologias. O *Oveia* é um extrator de ontologias representadas no formato Topic Maps. Sua arquitetura é composta por duas especificações e os referentes processadores: a primeira, escrita na linguagem XSDS (*XML Specification for DataSources/DataSets*), especifica os dados a serem extraídos das fontes de informação; enquanto que segunda, escrita na linguagem XS4TM (*XML Specification for Topic Maps*), é responsável por declarar as ontologias a serem geradas. Com base nestas especificações, o extrator busca as informações nas fontes de informação e produz um topic map. Este topic map gerado pode ser armazenado em formato XTM (XML Topic Maps) ou em uma base de dados relacional. Essa dupla capacidade de manipular vários tipos de fontes de informação e de poder armazenar o resultado em um suporte diferente é vantagem na comparação com as outras ferramentas de extração de ontologias; nomeadamente com o seu antecessor, o TM-Builder, que apenas permitia lidar com documentos XML, relativo ao qual este representa uma evolução justificativa.

Palavras chaves: Topic Maps, Extração de Ontologias, Semantic Web, XML, XSL.

Abstract

This paper presents a proposal based on ontology to achieve semantic interoperability in a heterogeneous information system. *Oveia* is an ontology extractor, following Topic Maps approach. *Oveia* was conceived to overcome the drawbacks of the known ontology extraction tools; namely, *Oveia* is a sucessor of *TM-Builder*. It provides an extraction model supported on an ontology specification language, XS4TM (*XML Specification for Topic Maps*) – a language to define the ontology to be extracted (topics, association, and instances) – but it also takes into consideration the characteristics of each data sources, interpreting a resource specification written in XSDS. The proposed extractor processes the XSDS and XS4TM specification and generates a topic map. This generated topic map can be stored in XTM syntax or in a relational database. This double capacity (to manipulate many kinds or information resources and to store the generated topic map in two different formats) is a clear advantage in comparison with the ontology extraction tools available.

Keywords: Topic Maps, Ontology Extraction, Semantic Web, XML, XSL.

[*]Bolsista CNPq - Brasil

1 Introdução

A chamada *Sociedade de Informação* necessita de ter acesso completo à informação disponível, a qual é geralmente heterogênea e distribuída. Para estabelecer uma partilha de informação eficiente, muitos problemas técnicos têm sido resolvidos. Primeiramente deve ser encontrada a fonte de informação apropriada (contendo os dados necessários para uma determinada tarefa). Encontrar recursos de informação apropriados é um problema tratado nas áreas de *recuperação de informação* (*information retrieval*) e *filtragem de informação* (*information filtering*) [2].
Uma vez localizada a informação, é necessário permitir o acesso aos dados. Isto significa que os recursos de informação, encontrados no primeiro passo, devem integrar e possibilitar que o sistema que efetuou a procura trabalhar com eles em conjunto. O problema de unir sistemas heterogêneos e sistemas distribuídos é conhecido como o *problema da interoperabilidade*.
Resumidamente, a partilha da informação não requer apenas que se tenha acesso completo aos dados; também requer que os dados acessados possam ser processados e interpretados pelo sistema remoto. Problemas que podem surgir da heterogeneidade dos dados são bem conhecidos na comunidade de sistemas de base de dados distribuídas (veja [7] e [6]): heterogeneidade estrutural (heterogeneidade esquemática) e heterogeneidade semântica (heterogeneidade de dados) [7]. Heterogeneidade estrutural significa que diferentes sistemas de informação armazenam seus dados em diferente estruturas. Heterogeneidade semântica resulta dos diferentes significados atribuídos a conteúdos semelhantes.
Para conseguir interoperabilidade semântica em sistemas heterogêneos de informação, o significado da informação que é intercambiada deve ser compreendido claramente pelo sistema interpretador [9]. Conflitos semânticos ocorrem toda vez que dois sistemas usam diferentes interpretações para a mesma informação, ou seja, quando há uma ambiguidade. Goh [4] identifica três causas principais para heterogeneidade semântica:

- Conflitos em geral ocorrem quando quando ítens de informação parecem ter o mesmo significado, mas de fato, diferem. Por exemplo, contextos temporais diferentes;
- Conflitos de escala ocorrem quando diferentes sistemas de referência são usados para medir um valor. Exemplo disto são diferentes moedas (valores monetários);
- Conflitos de nomes ocorrem quando os nomes em esquemas de informação diferem significativamente. Um fenômeno freqüente é a presença de homônimos e sinônimos.

O uso de ontologias para a explicação do conhecimento implícito e oculto é uma possível abordagem para resolver o problema da heterogeneidade semântica. Em [12], menciona-se a interoperabilidade como uma aplicação chave de ontologias, sendo ai referidas muitas abordagens baseadas em ontologias para integração de informação. É neste contexto que surge a nossa motivação para lidar com ontologias, tendo-se optado pela sua própria representação na forma de Topic Maps.
O processo de desenvolvimento de ontologias baseadas em Topic Maps é complexo, consumidor de tempo e requer grande quantidade de recursos humanos e financeiros, devido ao fato de qualquer topic map (por mais simples que seja) possuir um conjunto significativo de tópicos e associações; além disso, para que a ontologia extraída seja realmente significativa, pode envolver um grande número de recursos de informação que poderão ser de tipos diferentes.
Para resolver este problema, este artigo propõe um extrator de ontologias, chamado *Oveia*, que constrói topic maps a partir de recursos heterogêneos de informação, tais como bases de dados relacionais e documentos XML. Para isso, a ontologia a ser extraída é definida em uma linguagem de especificação denominada XS4TM (*XML Specification for Topic Maps*). O topic map extraído pode ser armazenado em uma base de dados relacional (permitindo que as ontologias possam crescer, sem restrições), ou em um documento no formato XML Topic Maps (XTM).
O artigo inicia apresentando o paradigma Topic Maps na seção 2; Topic Maps é um formalismo para representar conhecimento sobre um recurso de informação, organizando por tópicos. A descrição do sistema que se propõe, o extrator de Topic Maps a partir de recursos heterogêneos de informação – *Oveia* – é feita na seção 3. A definição da linguagem XS4TM será encontrada na seção 3.5. A seção 4 apresenta os trabalhos relacionados. Por fim, uma síntese do artigo e os trabalhos futuros são apresentados na conclusão.

2 Topic Maps

Topic Maps [3] é um formalismo para representar conhecimento acerca da estrutura de um conjunto de recursos de informação e para o organizar em *tópicos*. Esses tópicos têm ocorrências e associações que representam e definem relacionamentos entre os tópicos. A informação sobre cada tópico pode ser inferida ao examinar as associações e ocorrências ligadas ao tópico. Uma coleção desses tópicos e associações é chamada *topic map*. Também pode ser visto como um paradigma que permite organizar, manter e navegar pela informação, permitindo transformá-la em conhecimento. Falar sobre Topic Maps, é falar sobre estrutura de conhecimento.

Um mapa de tópicos expressa a opinião de alguém sobre o que os tópicos são, e quais as partes do conjunto de informação que são relevantes para cada tópico [11].
Permitindo a criação de um mapa virtual da informação, os recursos de informação mantém-se em sua forma original e não são modificados. Então, o mesmo recurso de informação pode ser usado de diferentes maneiras, por diferentes mapas de tópicos. Como é possível e fácil modificar um mapa, a reutilização da informação é conquistada.
Tópicos são o ponto principal de Topic Maps [10]. Em um sentido mais genérico, um tópico pode ser qualquer coisa: uma pessoa, uma entidade, um conceito. Eles constituem a base para a criação de Topic Maps (TM). Cada tópico tem um tipo de tópico (*topic type*), ou talvez múltiplos tipos. Cada tipo de tópico pode ser visto como uma típica relação classe-instância.
Ao analisar Topic Maps, identificam-se duas camadas distintas: os tópicos e as ocorrências. Os tópicos podem ser divididos em duas partes: os que representam conceitos abstratos e os que representam conceitos concretos. A ontologia é definida pelos conceitos abstratos, ou seja, os que serão instanciados por outros tópicos; por exemplo: tipo de tópico, tipo de associação e pelo tipo de papel de atuação em ocorrências.
Os tópicos restantes formam a base de conhecimento associada à ontologia, os quais compõem um conjunto de objetos de informação que permite organizar e indicar os reais recursos de informação (um objeto pode ter múltiplas ocorrências nos recursos de informação). A figura 1 dá uma representação esquematizada desta visão.

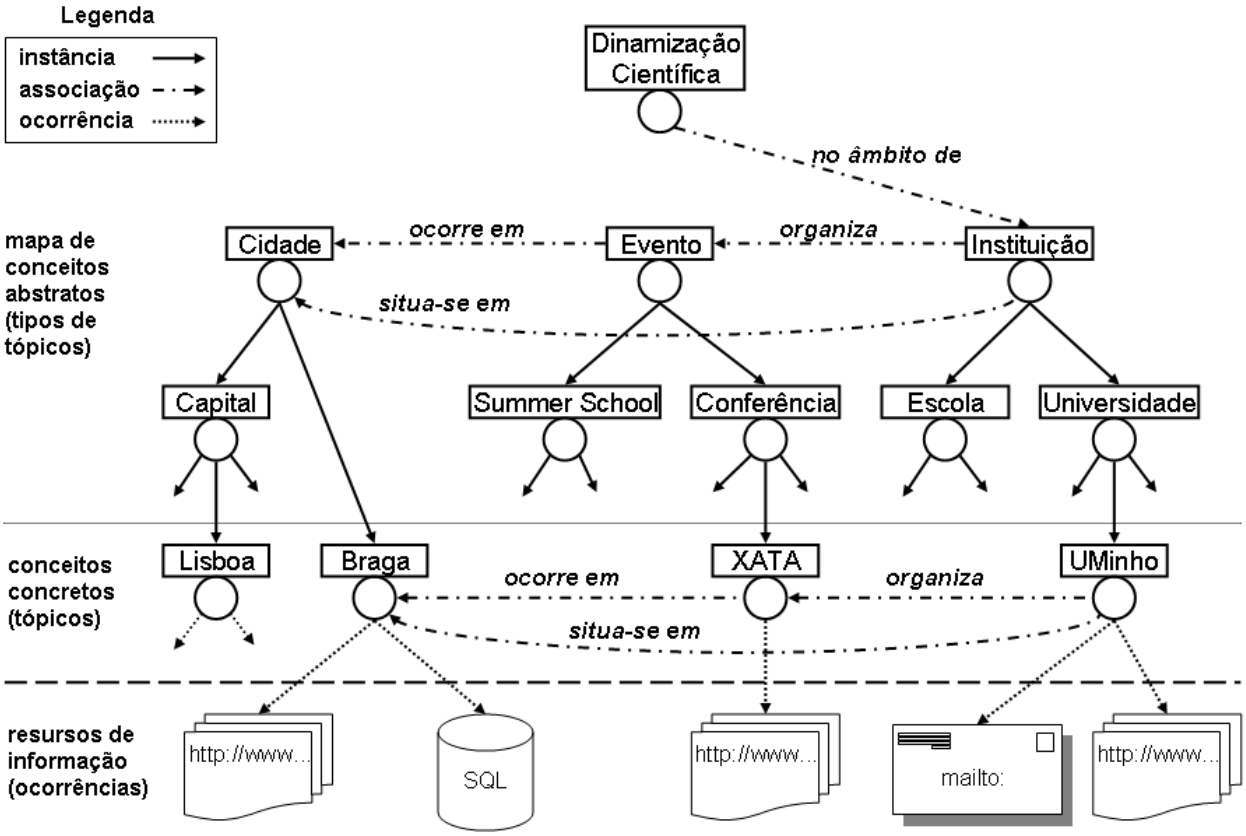


Figure 1: O Mapa do Conceito *Dinamização Científica*.

O conceito de associação (*association*) permite descrever relações entre tópicos. Uma associação é (formalmente) um elemento de vínculo que define uma relação entre dois ou mais tópicos. Um ilimitado número de tópicos podem ser relacionados por uma associação.

3 Oveia – Um Extrator de Topic Maps a partir de Recursos de Informação

O *Oveia* é um extrator de ontologias em sistemas heterogêneos de informação baseado em Topic Maps. O *Oveia* foi desenvolvido com o objetivo de suprir as deficiências encontradas nas atuais ferramentas de extração de ontologias. O *Oveia* surge na seqüencia do projeto que resultou no *TM-Builder* [8], o qual fornece um modelo de extração suportado numa linguagem própria para a especificação da ontologia a ser extraída.
Um fato que representa a evolução do *Oveia* em relação à sua versão inicial é a capacidade de extrair ontologias a partir de fontes de dados diversas. No *TM-Builder*, quando a fonte de informação é diferente de um documento XML, há uma necessidade uma conversão desta fonte para um arquivo XML. Portanto, considerando que a maior parte dos

recursos de informação em empresas e instituições estão armazenadas em base de dados, para realizar uma extração de uma ontologia a partir delas, inicialmente seria necessário a geração de documentos XML com o conteúdo da base de dados.

Assim como o *TM-Builder*, o *Oveia* faz uso de uma linguagem de especificação de extração (XS4TM), a qual permite extrair Topic Maps de forma genérica e adaptativa. A especificação de extração de ontologias em XS4TM tornou-se mais flexível e completa, contemplando todos os elementos do padrão Topic Maps [3]. Isso garante maior flexibilidade de especificação para propósitos diversos de extração.

Na fase de especificação dos recursos de informação, é possível fazer transformações e filtros nessas fontes de dados, pois é utilizada a linguagem de consulta de cada recurso para sua especificação.

A linguagem de especificação de extração foi inspirada no modelo XTM. Isso significa que a especificação da ontologia a ser extraída (em XS4TM) é feita em um esquema XML similar ao esquema de XTM. Essa característica permite maior facilidade de compreensão da especificação proposta, pois o modelo XTM é um padrão que vem sendo adotado amplamente pela comunidade acadêmica. Assim, o projetista da ontologia apenas deve conhecer a sintaxe de XTM e a estrutura das fontes de dados, para estar habilitado a especificar extrações de ontologias em XS4TM.

A arquitetura do *Oveia* pode ser expressa conforme demonstra a figura 2: inicialmente, é feita em uma especificação XSDS (*XML Specification for DataSources/DataSets*), a qual define quais dados que devem ser recuperados pelo *Extrator de Datasets*; a informação extraída é armazenada em um formato intermediário, chamado *Datasets*. O próximo passo é a especificação da ontologia em XS4TM; essa fase determina o que é relevante para a extração dos tópicos e associações, assim como clarifica os limites que devem ser impostos ao topic map. O processador XS4TM recebe os *datasets* gerados e a especificação da ontologia na linguagem XS4TM (*XML Specification for Topic Maps*) e gera o topic map final. Por fim, o *Oveia* armazena o topic map gerado na BD Ontologia ou no formato XTM.

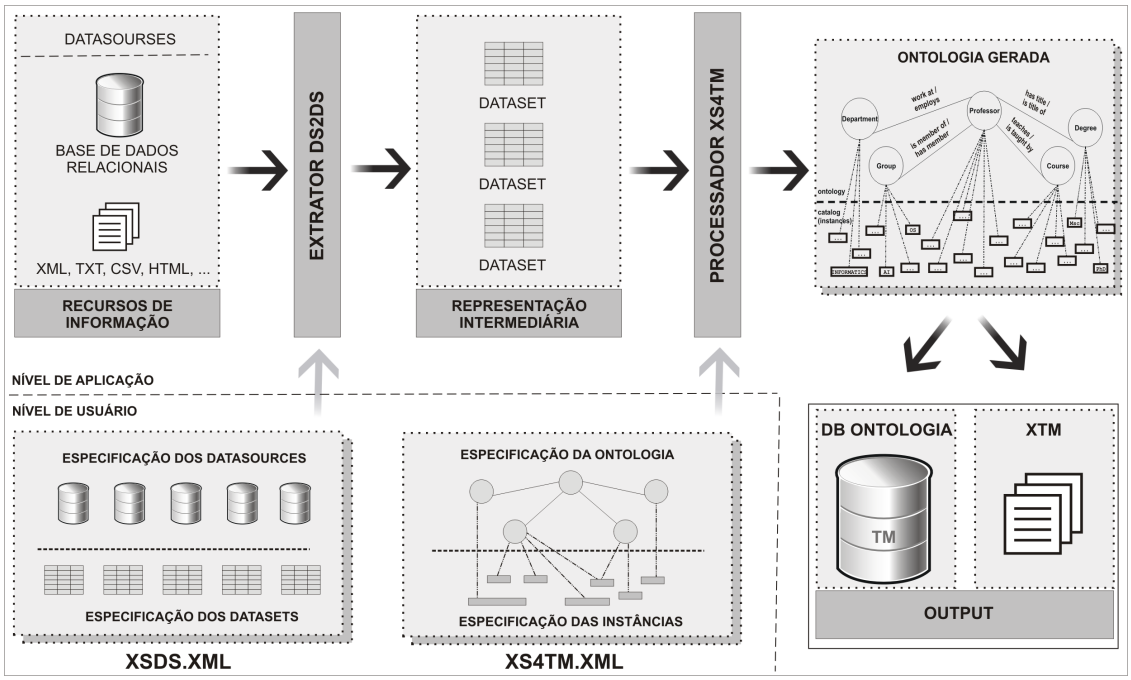


Figure 2: Arquitetura do Oveia

As próximas sub-seções apresentam cada um dos componentes do *Oveia*.

3.1 Recursos de Informação: Os *Datasources*

Este componente é composto pelos recursos de dados: bases de dados, documentos XML, páginas HTML, etc. Ao final do processo de extração de ontologias, os recursos manterão-se inalterados, ou seja, o *Oveia* não modifica as fontes; somente copia as partes relevantes de informação para a construção do topic map. Esses recursos de dados são mapeados para uma representação intermediária, chamada *datasets*. Esse mapeamento é descrito pela linguagem XSDS.

3.2 Representação Intermediária da Informação: Os *Datasets*

Os *datasets* são a representação intermediária que contém os dados extraídos das fontes de informação. Cada *dataset* tem uma relação com uma entidade dos *datasources*, e seu conteúdo é representado na forma de uma tabela, onde

cada linha é um registro segundo a estrutura definida em XSDS. Os *datasets* garantem que o *Oveia* tenha uma visão uniforme sobre a estrutura de dados que representam as fontes de dados participantes.
Cada *dataset* tem uma identidade única, a qual será usada pelo *Oveia* para o referenciar. A idéia fundamental é que todos os objetos tem rótulos que descrevem o seu conhecimento. Por exemplo, o seguinte objeto representa um registro da categoria de tipo de professor:

<1,PhD>

onde "1" é o identificador da categoria, enquanto que "PhD" é um rótulo legível por humanos. Os *datasets* são simples, enquanto provém um poder de expressividade e flexibilidade necessário para integrar sistemas de informação de diferentes fontes.

3.3 XSDS: Especificação das Fontes de Dados

XSDS (*XML Specification for DataSources/DataSets*) é a linguagem definida com o intuito de especificar quais fontes de informação fornecerão dados para a criação de topic maps, de acordo com uma ontologia posteriormente especificada. Essa especificação fornece todos os elementos necessários para especificar as fontes de dados passíveis de extração de informação.
No inicio deste trabalho foi previsto a criação de uma linguagem universal para realizar consultas em recursos heterogêneos de informação. Mas a conversão das linguagens de consulta (SQL, XPath, ...) para a linguagem idealizada como universal não seria a solução ideal pelo fato de que o projetista da ontologia teria a necessidade de aprender uma nova linguagem, ao invés de usar as linguagens amplamente conhecidas e difundidas. Dessa forma foi adotada a estratégia de manter o padrão de consulta de cada recurso e processar o resultado gerado por cada uma das linguagens de consulta. Assim, os `datasets` seriam o ponto de unificação de representação da informação entre os metadados extraídos de cada recurso de informação.
De um modo formal, a *Context Free Grammar* de XSDS é definida a seguir:

```
1 Resources   ::= Datasources Datasets
2 Datasources ::= extratorDriver name Datasource+
3 Datasource  ::= Parameter+
4 Parameter   ::= name parameter
5 Datasets    ::= Dataset+
6 Dataset     ::= name database dataset
```

Na prática, a gramática de XSDS é dividida em duas partes: a definição dos *datasources* e a definição dos *datasets*. A primeira parte refere-se aos recursos físicos, ou seja, define-se quais fontes reais de informação serão usadas para a obtenção de dados; a segunda parte refere-se a quais campos de dados das fontes de informação devem ser extraídos, usando a linguagem de query de cada fonte em questão. Assim, pode-se dizer que a partir de um mesmo *datasource*, podem ser construídos vários *datasets*.
A definição da arquitetura do extrator foi idealizada para suportar extensão a diversos recursos de informação como fontes de dados. Para isso, essa arquitetura baseia-se no conceito de *drivers* de extração.

3.3.1 Especificação dos Datasources e Datasets em XSDS:

Os *datasources* definem a localização física do recurso de informação. A declaração de cada uma das fontes de informação é feita no elemento <*datasource*>. Este elemento possui um atributo, chamado *extractorDriver* que indica qual *driver* de extração será utilizado: de acordo com o tipo de fonte de informação. Por exemplo: no caso de uma base de dados, além da localização da mesma, são passados parâmetros como o usuário e a senha a ser utilizada nesta base de dados, juntamente com o *driver* de extração que fará este processo; no caso da fonte de informação ser um documento XML, é necessário apenas o nome do arquivo com o seu caminho na árvore de diretórios do sistema operacional.
Para cada conjunto de dados dos recursos de informação (*datasets*) que se queira mapear a partir dos recursos de informação, é necessária a declaração do elemento <*dataset*>. Neste elemento, é necessário indicar qual fonte de dados provém os dados para a construção do *dataset* em questão.
O conteúdo do elemento <*dataset*> é uma expressão na linguagem de consulta referente ao tipo da fonte de informação. Caso esta fonte seja uma base de dados, o conteúdo deste elemento será uma expressão SQL para recuperar os dados referentes ao *dataset* em questão. Se a fonte de informação é um documento XML, o conteúdo deste elemento será uma expressão XPath, indicando o caminho para a informação deste *dataset*.

3.4 O Extrator DS2DS

O *Extrator DS2DS* (*DataSource to DataSet*) é um processador que extrai dados de recursos de informação e faz a criação dos *datasets*, de acordo com a especificação XSDS. Este componente processa uma especificação XSDS, a

qual especifica a fonte dos dados a serem extraídos (*datasources*) e o destino das informações extraídas, as quais definem a representação intermediária (*datasets*).
Esta representação intermediária é composta por um conjunto de tabelas que contém a informação extraída dos datasources. Estas tabelas contém somente os dados selecionados nos elementos *datasets* da especificação XSDS em questão.
O *Extrator DS2DS* possui diversos *drivers* de extração que, como dito anteriormente, são os módulos responsáveis pela extração de informação das fonte de dados; portanto, há um *driver* desenvolvido para cada tipo de recurso de informação.
Atualmente, o protótipo do Oveia possui dois drivers implementados: para conectar com base de dados relacionais e para recuperar informação de documentos XML. A implementação de novos *drivers* para outros recursos de informação é um processo relativamente fácil e pode ser realizado conforme a necessidade.

3.5 XS4TM: Uma linguagem XML para especificar a extração de Topic Maps

A linguagem XSTM, proposta em [8], foi inicialmente definida como sendo um dialeto XML para especificar o topic map que se pretende construir ao analisar documentos anotados pertencentes a um mesmo esquema XML. Por essa definição, o XSTM está diretamente ligado a extrações a partir de documentos XML. Por outro lado, a necessidade de abranger novas fontes de dados fez com que se propusesse uma nova arquitetura para extração de ontologias. Dessa forma tornou-se necessário repensar e redesenhar o XSTM; o qual passa a ser denominado por XS4TM.
Cada especificação XS4TM é uma instância XML. Portanto, na prática a linguagem XSTM é definida por um DTD (e/ou um XML-Schema), de modo a permitir o uso de todos os ambientes de processamento XML.
A linguagem XS4TM tem por objetivo tornar a especificação da extração de Topic Maps mais completa e flexível. XS4TM é caracterizado por transformar o atual padrão XTM em um subconjunto da sua especificação.
Formalmente, XS4TM é representado pela *Context Free Grammar* abaixo:

```
1  XS4TM           ::= Ontologies Instances
2  Ontologies      ::= (Topic | Association)*
3  Instances       ::= (Topic | Association)*
4  Topic           ::= id (InstanceOf* SubjectIdentity? (BaseName | Occurrence)*)
5  InstanceOf      ::= id (TopicRef | ResourceRef | SubjectIndicatorRef)
6  SubjectIdentity ::= id (TopicRef | ResourceRef | SubjectIndicatorRef)*
7  TopicRef        ::= id xlink:type xlink:href
8  SubjectIndicatorRef ::= id xlink:type xlink:href
9  BaseName        ::= id (InstanceOf? Scope? BaseNameString Variant*)
10 BaseNameString  ::= id baseNameString
11 Variant         ::= id (Parameters VariantName? Variant*)
12 VariantName     ::= id (ResourceRef | ResourceData)
13 Parameters      ::= id (TopicRef | ResourceRef | SubjectIndicatorRef)+
14 Occurrence      ::= id (InstanceOf? Scope? (ResourceRef | ResourceData))
15 ResourceRef     ::= id xlink:type xlink:href
16 ResourceData    ::= id resourceData
17 Association     ::= id (InstanceOf? Scope? Member+)
18 Member          ::= id (RoleSpec? (TopicRef | ResourceRef | SubjectIndicatorRef)*)
19 RoleSpec        ::= id (TopicRef | ResourceRef | SubjectIndicatorRef)
20 Scope           ::= id (TopicRef | ResourceRef | SubjectIndicatorRef)+
```

A especificação XS4TM é subdividida em duas partes distintas (ambas seguem o esquema de XTM 1.0 DTD):

- Na primeira parte são declarados os elementos responsáveis pela definição da ontologia, como os tipos de tópicos, os tipos de associações, ou qualquer outra definição de acordo com o modelo XTM que possa ser utilizada para expressar a estrutura de conhecimento a ser extraída – ou seja, a hierarquia de tópicos, relações super-tipo/sub-tipo e associações entre os tópicos;
- A segunda parte representa as instâncias de tópicos e associações. Nesse momento, os `Datasets` serão utilizados para expressar quais recursos de informação fornecerão metadados para a construção de tópicos e associações.

A figura 3 mostra uma referência da especificação XS4TM para um elemento do *dataset DS_aluno* declarado em XSDS.
Para o preenchimento das informações referentes a cada tópico, é necessário buscar tal informação no *dataset* que a contém. Assim, identifica-se essas propriedades com a expressão:

@ + ”dataset” + ”.” + ”atributo”

Mais detalhadamente, isto significa:

- O @ apenas indica que esta declaração é referente a uma propriedade de um *dataset*;

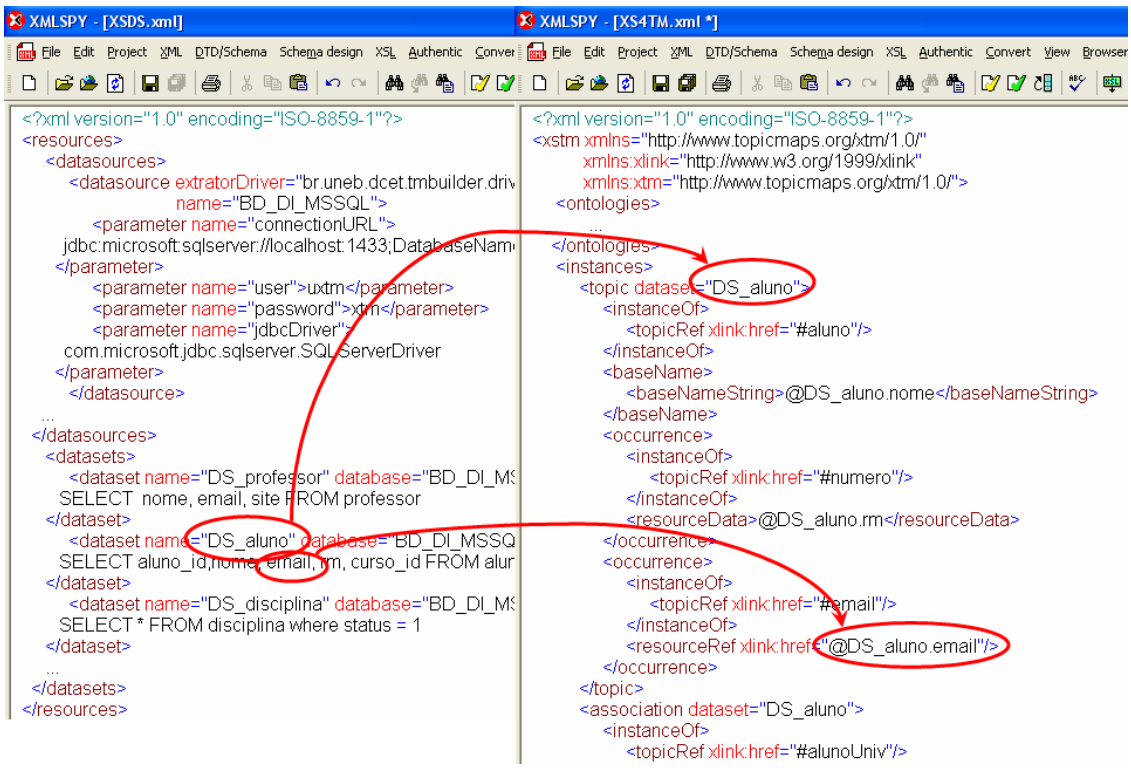


Figure 3: Relações entre XSDS e XS4TM

- Após o @, encontra-se o identificador do *dataset* (especificado em XSDS) ao qual deseja-se recuperar a informação. No exemplo da figura 3, o *dataset* selecionado é o *DS_aluno*.
- O *atributo* é uma referência ao campo do *dataset* que contém a informação desejada. Na figura 3, o atributo recuperado para a construção da ocorrência do tipo *email* é o campo *email* extraído pelo *dataset DS_aluno*.

Desta forma, cria-se uma forma de habilitar o uso das informações contidas nos *datasets*.

3.6 Processador de XS4TM

Este componente utiliza a especificação XS4TM para selecionar quais campos dos *datasets*, extraídos dos recursos de informação, são necessários para a formação do topic map. Este processador é um interpretador que tira vantagem da organização das informações em um formato uniforme.
O seu processo de execução pode ser resumido em três passos: (1) ler a especificação XS4TM e extrair os dados especificados que encontram-se nos *datasets*; (2) criar o topic map baseado na própria especificação XS4TM; (3) armazenar o topic map gerado na BD Ontologia ou em um documento no formato XTM.

3.7 Base de Dados de Ontologias

Um dos diferenciais desta ferramenta é o armazenamento dos Topic Maps extraídos em uma base de dados relacional. De acordo com [15], referente aos métodos de mapeamento de documentos XML para o modelo relacional, adotou-se na *BD Ontologia* o modelo de mapeamento por estrutura.
Conforme o mapeamento por estrutura, foi criada uma tabela para cada elemento de XTM 1.0 DTD. Esse processo consiste em identificar as características e os tipos de associações entre os elementos do DTD e representa-los no modelo relacional.
Pode ser entendido facilmente tomando um trecho desse mapeamento, como apresenta o segmento do XTM 1.0 DTD abaixo e a figura 4.

```
<!ELEMENT topic (instanceOf*, subjectIdentity?, (baseName | occurrence)*)>
<!ATTLIST topic
    id ID #REQUIRED
>
<!ELEMENT baseName (scope?, baseNameString, variant*)>
<!ATTLIST baseName
    id ID #IMPLIED
>
```

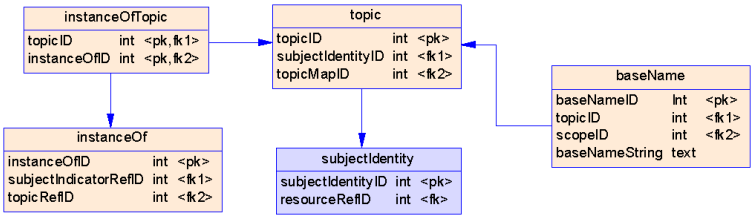


Figure 4: Trecho do Modelo ER do BD Ontologia

A facilidade de compreensão desse modelo é garantida principalmente pelo fato de que este modelo segue o padrão XTM, o qual é bastante conhecido pela comunidade acadêmica. Essa foi umas das vantagens trazidas por essa opção de modelagem, preservando o padrão Topic Maps. Assim, a partir dessa base de dados, é possível navegar no Topic Maps utilizando consultas SQL.

4 Trabalhos Relacionados

Vários projetos tem objetivos similares ao *Oveia*. O TSIMMIS [1] é um projeto que objetiva prover ferramentas para acessar, de uma maneira integrada, múltiplos recursos de informação e garantir que a informação obtida é consistente. O TSIMMIS foi desenvolvido para extrair propriedades de objetos não-estruturados, permitindo a combinação entre várias fontes e a navegação nesta informação; ele fornece uma visão centralizada da informação que esta dispersa nos recursos de informação. O *Oveia* foi desenvolvido para permitir uma navegação conceitual sobre recursos heterogêneos de informação. Essa navegação conceitual é obtida por uma ontologia, representada de acordo com a norma Topic Maps, criada a partir de dados extraídos dos sistemas de informação.
O KAON[1] [5] é um projeto *open-source* que fornece uma infra-estrutura para gestão de ontologias, voltado para aplicações de negócios. A ferramenta KAON REVERSE é um *plug-in* do *framework* KAON. O KAON REVERSE permite o mapeamento de bases de dados relacionais para uma ontologia, com o objetivo de extrair instâncias e relacionamentos entre instâncias, a partir da base de dados. Entre as ferramentas conhecidas, esta é a que mais se aproxima do *Oveia*.
A tabela 1 mostra as características e funcionalidades do *Oveia* e do KAON REVERSE[2].

	KAON REVERSE	OVEIA
Linguagem	Java	Java
Uso de APIs	Sim	Sim
Uso de Engenharia Reversa	Sim	Não
Especificação	Árvore(GUI)	Documento XML
Fontes de Extração de Ontologias	BDs relacionais via JDBC	BDs relacionais via JDBC, XML, extensível a outras fontes
Padrão de Representação de Ontologias	RDF	Topic Maps
GUI (Interface Gráfica)	Sim	Não
Resultado Gerado	Documento RDF	Base de Dados de Ontologias

Table 1: Comparativo entre KAON REVERSE e Oveia.

De acordo com a tabela 1, é perceptível as vantagens de cada ferramenta em pontos distintos. Seria mais sensato dizer que existe uma certa tendência à união das funcionalidades, do que comparar qual seria a melhor ferramenta. Partindo deste ponto de vista, destaca-se as vantagens de cada uma. Por um lado, a KAON REVERSE apresenta vantagens em relação ao uso de interface gráfica para a especificação de ontologias e o uso de engenharia reversa dos recursos de informação para auxiliar o mapeamento. Por outro lado, o *Oveia* se destaca por ser mais flexível em relação aos recursos passíveis de extração e em relação ao processo de especificação. Além disso, o *Oveia* se diferencia por gerar uma base de dados de ontologias capaz de manter os dados extraídos.
Em relação ao padrão de representação, as duas ferramentas possuem focos diferentes. Enquanto uma faz uso de RDF, a outra utiliza Topic Maps. A opção pelo paradigma Topic Maps foi tomada, principalmente, por seu poder de representatividade ser superior ao do RDF e permitir a criação de navegadores conceituais.
Ao fazer uso de uma linguagem de especificação semelhante ao padrão Topic Maps (XS4TM), o *Oveia* flexibiliza o processo de extração para o projetista da ontologia. Esta vantagem é considerável quando se refere ao processo

[1]Mais informações em: http://kaon.semanticweb.org/
[2]Os dados presentes nesta tabela foram utilizados a partir da análise feita em [13].

de criação de ontologias, visto que este é diretamente ligado a sua finalidade. Ou seja, para cada caso, poderá ser criada uma visão diferente da fonte de dados, expressando assim uma estrutura de conhecimento específica sobre um determinado domínio.
Um outro extrator de ontologias que pode ser referido é o *OntoExtract* [14]. O *OntoExtract* extrai ontologias a partir do esquema de bases de dados relacionais. Este extrator apresenta a proposta de definir o RDF como linguagem base para definição de objetos de ontologias, como uma espécie de camada de modelo; enquanto que o XTM seria utilizado como uma linguagem de definição de ontologias, uma (meta)ontologia.
Assim como o KAON REVERSE, o *OntoExtract* baseia-se na extração exclusiva em bases de dados a partir de engenharia reversa do esquema relacional. Limitando, assim, a faixa de recursos de informação passíveis de extração.

5 Conclusão

O objetivo deste artigo foi a apresentação de uma arquitetura para a construção automática de Topic Maps, para extração de informação de bases de dados ou documentos XML. Esse sistema, designado por *Oveia*, resultou de uma proposta inicial denominada *TM-Builder*, o qual é um extrator de ontologias a partir das fontes XML totalmente escrito em XML/XSL.
A extração de informação de fontes heterogêneas de informação é especificada pela linguagem XS4TM, a qual define quais os conceitos e relações encontradas nestas fontes de informação que serão mapeadas para tópicos e associações, respectivamente, no topic map gerado pelo *Oveia*. Para funcionar, o sistema requer que todas as fontes estejam descritas na linguagem XSDS, de modo a ser convertidos para uma representação interna uniforme, os *Datasets*.
XS4TM é a linguagem para especificar a extração de Topic Maps a partir de recursos de informação. XS4TM classifica os tópicos, dando-lhes uma semântica mais concreta, através da associação de um tipo de tópico, um tipo de associação ou um tipo de papel de atuação em associações (valores encontrados nas fontes).
O *Oveia* é uma evolução do *TM-Builder*, o qual estava limitado ao processamento de documentos XML. A fim de evitar esse problema, foi construída uma nova arquitetura que fornece uma abstração dos tipos de fontes de dados baseada em *drivers* de extração. O resultado obtido foi a introdução de uma representação intermediária, os referidos *Datasets* que asseguram independência da fonte de dados através da criação de uma nova camada de abstração, o que torna possível a extração em fontes de dados diversas, de forma transparente e em uma única especificação.
A principal vantagem desta proposta é a possibilidade de adaptação do processo de especificação e extração de ontologias a um maior número de casos reais, sem acarretar mudanças de formato dos recursos de informação.
Outra importante vantagem surge a nível de manutenção dos topic maps: modificações nas fontes de informação (obviamente mudanças ao nível de seu conteúdo, e não estruturais – incluindo-se novos dados ou excluindo-os), não implicam uma modificação da especificação XS4TM; basta voltar a executar o processo de extração sobre a fonte de informação alterada para obter um novo mapa de tópicos atualizado.
Um dos projetos a ser desenvolvido em breve será um módulo que permita a conversão de um documento XTM para uma BD Ontologia, assim como possibilite a extração de um topic map da BD Ontologia para um documento XTM. Com este módulo, o utilizador pode rapidamente ter um conjunto de topic maps armazenados no formato desejado, seja em documentos XML (no padrão XTM) ou em base de dados relacionais (em uma BD Ontologia).
O *Oveia* não possui um ambiente amigável para o utilizador criar suas especificações. Aparentemente o processo de criação de um documento XS4TM ou XSDS mostra-se trabalhoso. Isso exige que o utilizador conheça a linguagem de descrição dos documentos de especificação, que é o padrão XTM. É sabido que esta tarefa pode ser auxiliada por ferramentas de edição de documentos XML, como por exemplo o XMLSpy[3], porém essa tarefa de criar uma interface de especificação agradável e fácil de usar será um outro tópico de investigação futura.

References

[1] TSIMMIS – The Stanford-IBM Manager of Multiple Information Sources, April, 1998. http://www-db.stanford.edu/tsimmis/tsimmis.html.

[2] N.J. Belkin and B.W. Croft. Information filtering and information retrieval: Two sides of the same coin? In *Communications of the ACM*, volume 35(12), pages 29–38. ACM, http://www-agki.tzi.de/buster/IJCAIwp/Finals/wache.pdf, December 1992.

[3] Michel Biezunsky, Martin Bryan, and Steve Newcomb. ISO/IEC 13250 - Topic Maps. ISO/IEC JTC 1/SC34, December, 1999. http://www.y12.doe.gov/sgml/sc34/document/0129.pdf.

[4] Cheng Hian Goh. *Representing and Reasoning about Semantic Conflicts in Heterogeneous Information Sources*. PhD thesis, Phd, MIT, 1997.

[3] Ferramenta para construção de documentos XML, desenvolvido pela ALTOVA. Mais informações em: http://www.altova.com

[5] S. Handschuh, A. Maedche, L. Stojanovic, and R. Volz. KAON, 2001. http://kaon.semanticWeb.org/kaon/white-paper.pdf.

[6] V. Kashyap and A. Sheth. Schematic and semantic semilarities between database objects: A context-based approach. In *The International Journal on Very Large Data Bases*, volume 5(4), pages 276–304. VLDB Journal, 1996.

[7] Won Kim and Jungyun Seo. Classifying schematic and data heterogeinity in multidatabase systems. In *IEEE Computer*, volume 24(12), pages 12–18. IEEE, 1991.

[8] Giovani Librelotto, José C. Ramalho, and Pedro R. Henriques. TM-Builder: Um Construtor de Ontologias baseado em Topic Maps. In *XXIX Conferencia Latinoamericana de Informtica – CLEI, La Paz, Bolívia*, 2003.

[9] Steven R. Newcomb. A Semantic Integration Methodology. In *Extreme Markup Languages 2003: Proceedings*. IDEAlliance, 2003. http://www.idealliance.org/papers/extreme03/html/2003/Newcomb01/EML2003Newcomb01.html.

[10] Jack Park and Sam Hunting. *XML Topic Maps: Creating and Using Topic Maps for the Web*. Prentice Hall, 2003.

[11] Steve Pepper. The TAO of Topic Maps - finding the way in the age of infoglut. Ontopia, 2000. http://www.ontopia.net/topicmaps/materials/tao.html.

[12] M. Uschold and M. Grniger. Ontologies: Principles, methods and applications. In *Knowledge Engineering Review*, volume 11(2), pages 93–155. Cambridge University Press, 1996.

[13] André Accioly Vieira. Extra o de Ontologias a partir de Esquemas Relacionais, 2002. Instituto Militar de Engenharia. Mestrado em Sistemas e Computação.

[14] André Accioly Vieira, Astério Kiyoshi Tanaka, and Ana Maria de Carvalho Moura. OntoExtract – Ferramenta para Extração de Ontologias a partir de Bancos de Dados Relacionais. *WTDBD/SBBD/2002*, 2002.

[15] Kevin Williams, Michael Brundage, Patrick Dengler, Jeff Gabriel, Andy Hoskinson, Michael Kay, Thomas Maxwell, Marcelo Ochoa, Johnny PaPa, and Mohan Vanmane. *Professional XML Databases*. Wrox Press, 2000.

Modelo de Referência para Melhoria de Processo de Software: uma abordagem brasileira

Kival C. Weber[1], Ana Regina Rocha[2], Ângela Alves[3], Arnaldo M. Ayala[1], Austregésilo Gonçalves[4] , Benito Paret[5] , Clênio Salviano[3] ,Cristina F. Machado[6], Danilo Scalet[6] , Djalma Petit[1] , Eratóstenes Araújo[1], Márcio Girão Barroso[1], Kathia Oliveira[7], Luiz Carlos A. Oliveira[6], Márcio P. Amaral[5], Renata Endriss C. Campelo[8], Teresa Maciel[8]

[1]Sociedade SOFTEX – Sociedade para Promoção da Excelência do Software Brasileiro
{ayala, dpetit, era, kival.weber, mgirao}@nac.softex.br

[2]COPPE/UFRJ – Universidade Federal do Rio de Janeiro
darocha@cos.ufrj.br

[3]CenPRA - Centro de Pesquisas Renato Archer
{angela.alves, clenio.salviano}@cenpra.gov.br

[4]Núcleo SOFTEX de Campinas
austre@cps.softex.br

[5]RIOSOFT – Núcleo SOFTEX do Rio de Janeiro
{benito, mpamaral}@riosoft.softex.br

[6]CELEPAR – Companhia de Informática do Paraná
{cristina, danilo, lcarlos}@pr.gov.pr

[7]UCB - Universidade Católica de Brasília
kathia@ucb.br

[8]CESAR - Centro de Estudos e Sistemas Avançados de Recife
{renata.endriss, teresa}@cesar.org.br

Resumo

Estudos sobre a qualidade no setor de software brasileiro mostraram a necessidade de um esforço significativo capaz de aumentar a maturidade dos processos de software das empresas brasileiras. Este artigo descreve o Projeto mps Br, uma iniciativa envolvendo universidades, grupos de pesquisa e empresas, sob coordenação da Sociedade SOFTEX (Sociedade para Promoção da Excelência do Software Brasileiro). Fundamentalmente, o projeto visa a criação e disseminação do Modelo de Referência para melhoria de processo de software (MR mps). Não é objetivo deste projeto definir algo novo no que se refere a normas e modelos de maturidade. A novidade do projeto está na estratégia adotada para sua implementação, criada para a realidade brasileira. O Modelo de Negócio definido para o projeto tem grande potencial de replicabilidade no Brasil e em outros países de características semelhantes, como por exemplo os países latinoamericanos.

1 Introdução

Pesquisas periódicas realizadas sobre a qualidade no setor de software mostram que é necessário um esforço concentrado para melhorar os processos de software no Brasil. Desde 1993, com a criação do PBQP Software (Subcomitê de Software do Programa Brasileiro da Qualidade e Produtividade), o Brasil investe na melhoria da Qualidade de Software (Weber, 1995, Weber, 2001). Entretanto, um estudo comparativo do MIT (*Massachussetts Institute of Technology*) (Veloso, 2003) constatou que realmente houve interesse na melhoria de processos de software no Brasil, nos últimos anos, mas que a empresa local favoreceu a ISO 9000 (ISO, 2000) em detrimento de outros modelos e padrões especificamente voltados para software. Segundo dados do MCT (Ministério da Ciência e Tecnologia), em 2003, o número de empresas que desenvolvem software no Brasil com certificação ISO 9000 era 214, enquanto o número de empresas com avaliação oficial CMM era 30.

Considerando-se estas 30 empresas, verifica-se que na base da pirâmide encontram-se 24 empresas no nível 2 e cinco empresas no nível 3. No topo da pirâmide há uma empresa no nível 4 e nenhuma no nível 5. Estes dados evidenciam que, para a melhoria dos processos de software no Brasil, há dois problemas a resolver nos próximos anos:

(i) no topo da pirâmide, a questão a ser resolvida é: *Como aumentar expressivamente o número de empresas com avaliação formal CMM/CMMI níveis 4 e 5 no Brasil, com foco nas empresas exportadoras de software e em outras grandes empresas?*

(ii) na base da pirâmide, tem-se outra questão que exige uma resposta: *Como melhorar radicalmente os processos de software no Brasil, com foco em um número significativo de micro, pequenas e médias empresas, de forma que estas atinjam os níveis de maturidade 2 e 3, a um custo acessível?*

A solução para o primeiro problema está fora do escopo deste trabalho, exigirá um prazo longo (4 a 10 anos) e será apoiada pelo Projeto Qualificação de Profissionais no Modelo CMMI. Este trabalho descreve uma abordagem para solução do segundo problema, no período 2004-2006, no âmbito do Projeto mps Br – Melhoria de Processo do Software Brasileiro. Os dois projetos são coordenados pela Sociedade SOFTEX (Sociedade para Promoção da Excelência do Software Brasileiro).

A Sociedade SOFTEX é uma entidade privada, sem fins lucrativos, que promove ações de empreendedorismo, capacitação, apoio à capitalização e ao financiamento, e apoio à geração de negócios no Brasil e no exterior, visando aumentar a competitividade da indústria brasileira de software. Sua missão é transformar o Brasil em um centro de excelência mundial na produção e exportação de software. A Sociedade SOFTEX é a entidade nacional responsável pelo Programa SOFTEX, que coordena as ações de 31 Agentes SOFTEX, localizados em 23 cidades de 13 Unidades da Federação, com mais de 1300 empresas associadas (consulte www.softex.br).

Após esta seção introdutória, na seção 2 é descrito o Projeto mps Br. Na seção 3 são resumidas as principais abordagens para melhoria de processo, que foram a base para a definição do Modelo de Referência (MR mps). A seção 4 apresenta o MR mps. A seção 5 descreve experiência-piloto em andamento e os próximos passos do projeto. Na seção 6, como conclusão, são destacadas as principais lições aprendidas até o momento e os diferenciais do projeto.

2 O Projeto mps Br

Em 2003, a Qualidade tornou-se uma das prioridades da Sociedade SOFTEX, elencada como um dos seus Projetos Estruturantes. Desde dezembro de 2003, sete renomadas instituições brasileiras, com competências complementares na melhoria de processos de software em empresas, participam do projeto Melhoria do Processo de Software Brasileiro (mps Br): a Sociedade SOFTEX, coordenadora do projeto; três instituições de ensino, pesquisa e centros tecnológicos (COPPE/UFRJ, CESAR, CenPRA); uma sociedade de economia mista, a Companhia de Informática do Paraná (CELEPAR), hospedeira do Subcomitê de Software da Associação Brasileira de Normas Técnicas (ABNT); e duas organizações não-governamentais integrantes do Programa SOFTEX (Sociedade Núcleo de Apoio à Produção e Exportação de Software do Rio de Janeiro – RIOSOFT e Sociedade Núcleo SOFTEX 2000 de Campinas). Desde o início do projeto, a COPPE/UFRJ convidou a Universidade Católica de Brasília (UCB) para ser sua parceira no projeto, que assim se uniu ao grupo.

O Projeto mps Br visa a melhoria de processos de software em empresas brasileiras, a um custo acessível, especialmente na grande massa de micro, pequenas e médias empresas. Tem como objetivo principal definir e implementar o Modelo de Referência para melhoria de processo de software (MR mps) em 120 empresas, até junho de 2006, com perspectiva de mais 160 empresas nos dois anos subsequentes. O projeto tem como objetivos secundários disseminar, em diversos locais no país: a capacitação no uso do modelo (cursos de Introdução ao MR mps e cursos e provas para Consultores de Implementação e Avaliadores do modelo); o credenciamento de instituições implementadoras e/ou avaliadoras do modelo, especialmente instituições de ensino e centros tecnológicos; a implementação e avaliação do modelo com foco em grupos de empresas.

O projeto compreende seis etapas. A 1ª etapa, finalizada em março de 2004, teve como objetivo organizar o projeto, estabelecer seus objetivos e definir a primeira versão do Modelo de Referência (MR mps). A 2ª etapa, concluída em junho de 2004, teve como objetivos o aprimoramento do Modelo de Referência, o início das atividades de treinamento no modelo e a realização de experiências iniciais de uso do MR mps em empresas. A partir de julho de 2004, em quatro etapas semestrais, estão sendo realizadas implementações em outras empresas, em diversos locais (especialmente onde houver Agente SOFTEX interessado e instituições credenciadas para implementação e/ou avaliação do modelo).

Normalmente, a implementação e avaliação de modelos como o CMMI, mesmo nos seus níveis mais baixos (2 e 3), está fora do alcance da micro, pequena e média empresa, especialmente no Brasil, devido ao seu custo elevado. Para resolver este problema, o Projeto mps Br criou dois modelos: (i) o Modelo de Referência para melhoria de processo de software (MR mps), que será descrito na seção 3, e, (ii) o Modelo de Negócio para melhoria de processo de software (MN mps), descrito nesta seção.

No Brasil, algumas instituições e um bom número de Agentes SOFTEX têm experiência na formação e gestão de grupos de empresas para melhoria de processo de software, seja de grupos de empresas voltados à implementação e certificação ISO 9000 (Weber et al, 1997) seja de grupos de empresas voltados à implementação e avaliação CMM e CMMI. A partir destas experiências concebeu-se para o projeto mps Br um modelo de negócios que prevê duas situações:

(iii) a implementação do MR mps de forma personalizada para uma empresa (MNE – Modelo de Negócio Específico);
(iv) a implementação do MR mps de forma cooperada em grupos de empresas (MNC – Modelo de Negócio Cooperado), com custo mais acessível às micro, pequenas e médias empresas por dividir proporcionalmente parte dos custos entre as empresas e por se buscar outras fontes de financiamento.

Para a implementação do MR mps e a realização de avaliações segundo o modelo, existirão instituições credenciadas. O credenciamento será feito pelo Forum de Credenciamento e Controle (FCC) do projeto. No momento do credenciamento, a Sociedade SOFTEX sempre assina um convênio com as instituições credenciadas. Os procedimentos e condições para o credenciamento serão descritos na seção 4 ao detalharmos o Modelo de Referência.

A Figura 1 ilustra o Modelo de Negócio definido para o Projeto mps Br. No Modelo de Negócio Específico para uma empresa (MNE), cada empresa interessada negocia e assina um contrato específico com uma das Instituições Credenciadas para Implementação (ICI) do MR mps. Para avaliação, negocia e assina um outro contrato específico com uma das Instituições Credenciadas para Avaliação (ICA). A entidade coordenadora do Projeto mps Br (Sociedade SOFTEX) toma conhecimento, através da ICI ou ICA, do contrato e dos resultados da implementação e/ou avaliação na empresa.

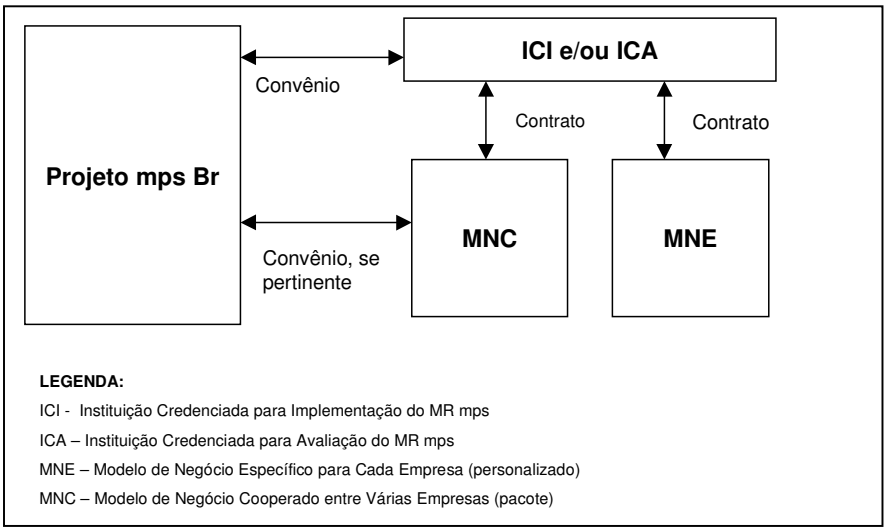


Figura 1 - Modelo de Negócio para melhoria de processo de software (MN mps)

No Modelo de Negócio Cooperado (MNC), o primeiro passo, é a constituição de um grupo de empresas interessadas na implementação do MR mps (o que pode acontecer, por exemplo, por iniciativa de um Agente SOFTEX). A partir de sua constituição, a coordenação do grupo de empresas irá negociar e assinar um contrato com uma das Instituições Credenciadas para Implementação (ICI) do MR mps. Posteriormente, irá negociar e assinar um outro contrato para avaliação das empresas com uma das Instituições Credenciadas para Avaliação (ICA). Neste caso, a Sociedade SOFTEX toma conhecimento da implementação e/ou avaliação no grupo de empresas, através da ICI ou ICA, e assina um convênio com a entidade organizadora do grupo de

empresas (por exemplo, um Agente SOFTEX), sempre que pertinente. Assim, a Sociedade SOFTEX e seus Agentes Regionais estarão acelerando a velocidade de implementação da melhoria de processos de software no Brasil, especialmente na micro, pequena e média empresa.

3 Modelos e Normas de Processo de Software

Fundamentalmente, o Projeto mps Br visa a criação e disseminação do Modelo de Referência para melhoria de processo de software (MR mps). Não é objetivo do projeto definir algo novo no que se refere a Normas e Modelos de Maturidade. A novidade do projeto está na estratégia adotada para sua implementação, criada para a realidade brasileira. Além disto, o Modelo de Negócio definido para o projeto tem grande potencial de replicabilidade no Brasil e em outros países de características semelhantes, como por exemplo os países latinoamericanos. Desta forma modelos, normas e métodos já disponíveis foram ponto de partida para a definição do Modelo de Referência (Figura 2)

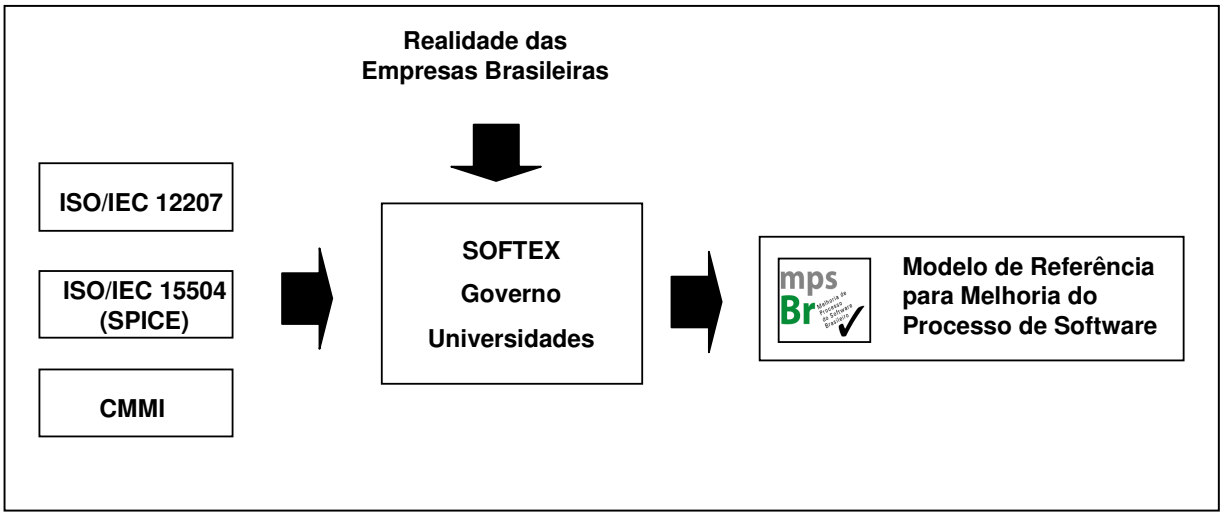


Figura 2 – Definição do Modelo de Referência

O ponto de partida para definição do MR mps foi, então, a análise da realidade das empresas brasileiras, a norma ISO/IEC 12207, a série de normas ISO/IEC 15504 (SPICE) e o modelo CMMI (*Capability Maturity Model Integration*), que descrevemos sucintamente a seguir.

Em 1988, foi proposto o desenvolvimento da Norma ISO/IEC 12207 (NBR ISO/IEC 12207, 1998) dentro de um esforço conjunto da ISO – *International Organization for Standardization* e do IEC – *International Electrotechnical Commission* em estabelecer uma estrutura comum para os processos de ciclo de vida de software como forma de ajudar as organizações a compreenderem todos os componentes presentes na aquisição e fornecimento de software e, assim, conseguirem firmar contratos e executarem projetos de forma mais eficaz. A Norma foi publicada internacionalmente em 1995 e no Brasil em 1998.

Em 2002, foi feita uma atualização na norma ISO/IEC 12207 (ISO/IEC PDAM 12207, 2002) em forma de anexo que visava representar a evolução da engenharia de software, as necessidades

vivenciadas pelos usuários da norma e a harmonização com a série de normas ISO/IEC 15504 – Avaliação de Processos de Software. Essa atualização inseriu processos e acrescentou na sua descrição propósito e resultados de implementação o que possibilita a avaliação da capacidade do processo. A norma, incluindo o seu anexo, é composta por 22 processos, 95 atividades, 325 tarefas e 254 resultados. Todos esses processos, executados durante o projeto de software, conduzem à qualidade tanto do produto quanto do processo. Entretanto, a norma deixa para a organização definir "como" os processos serão executados conservando dessa forma a flexibilidade necessária para que os países e as organizações a implementem de acordo com a cultura local e a tecnologia empregada.

A ISO/IEC 12207 tem sido amplamente utilizada em muitos países como referência para contratação de serviços de desenvolvimento e manutenção de software. No Brasil, muitas organizações têm tomado conhecimento da existência da norma e algumas já a utilizam para definição de processos de desenvolvimento de software. Muitos trabalhos de pesquisa têm utilizado a norma o que vislumbra uma ampla utilização da mesma no futuro.

Em setembro de 1992, a ISO realizou um estudo chamado "Necessidades e Exigências para uma Norma de Avaliação de Processos de Software". O trabalho concluiu que era pertinente a elaboração de um padrão que fosse aplicável à melhoria de processos e à determinação da capacidade. Este padrão deveria considerar os métodos e normas já existentes (como por exemplo, o CMM e a ISO 9001), abranger todos os processos de software e ser construído pelos especialistas que já desenvolviam e trabalhavam com os métodos e normas existentes à época. Como resultado desse primeiro trabalho, a ISO iniciou em janeiro de 1993 o projeto SPICE (*Software Process Improvement and Capability dEtermination*) com o objetivo de produzir inicialmente um Relatório Técnico que fosse, ao mesmo tempo, mais geral e abrangente que os modelos existentes e mais específico que a norma ISO 9001 (Salviano, 2001). Uma versão do SPICE foi aprovada em 1998 como Relatório Técnico e, em 2003, foi publicada a Norma ISO/IEC 15504 (ISO/IEC, 2003).

A ISO/IEC 15504 (SPICE) presta-se à realização de avaliações de processos de software com dois objetivos: a melhoria de processos e a determinação da capacidade de processos de uma organização. Se o objetivo for a melhoria de processos, a organização pode realizar a avaliação gerando um perfil dos processos que será usado para a elaboração de um plano de melhorias. A análise dos resultados identifica os pontos fortes, os pontos fracos e os riscos inerentes aos processos. No segundo caso, a empresa tem o objetivo de avaliar um fornecedor em potencial, obtendo o seu perfil de capacidade. O perfil de capacidade permite ao contratante estimar o risco associado à contratação daquele fornecedor em potencial para auxiliar na tomada de decisão de contratá-lo ou não (Salviano, 2001, ISO/IEC, 2003)

O modelo SW-CMM (*Capability Maturity Mo*del) foi definido no SEI (*Software Engineering Institute*) a pedido do Departamento de Defesa dos Estados Unidos. A partir de 1991, foram desenvolvidos CMMs para várias disciplinas (engenharia de sistemas, engenharia de software, aquisição de software, gerência e desenvolvimento da força de trabalho, desenvolvimento integrado do processo e do produto). Embora estes modelos tenham mostrado sua utilidade, o uso de múltiplos modelos se mostrou problemático. O CMMI surgiu para resolver o problema de se usar vários modelos e é o resultado da evolução do SW-CMM, SECM (*System Engineering Capability Model*) e IPD-CMM (*Integrated Product Development Capability Maturity Model*). É,

portanto, o sucessor destes modelos. Além disso o CMMI foi desenvolvido para ser consistente e compatível com a ISO/IEC 15504 (Chrissis, 2003).

Existem dois tipos de representação no CMMI: em estágios e contínua. Tem-se, assim, um único modelo que pode ser visto de duas perspectivas distintas. A representação em estágios é a representação usada no SW-CMM. Esta representação define um conjunto de áreas de processo para definir um caminho de melhoria para a organização, descrito em termos de níveis de maturidade. A representação contínua é o enfoque utilizado no SECM, no IPD-CMM e também no SPICE. Este enfoque permite que uma organização selecione uma área de processo específica e melhore com relação a esta área. A representação contínua usa níveis de capacidade para caracterizar melhoria relacionada a uma área de processo.

Uma questão que se apresenta para as organizações é, então: que modelo escolher? Se a organização é familiar com o SW-CMM, a representação em estágios será a mais adequada para migrar para o CMMI. Esta representação, também, é a mais adequada se a organização necessita demonstrar externamente o seu nível de maturidade. Entretanto, não há obrigatoriedade de se escolher uma representação em detrimento da outra. Mais de 80% do conteúdo das duas representações é comum e elas oferecem resultados equivalentes. Raramente as organizações implementam uma representação exatamente como ela é prescrita. Por exemplo, uma organização pode escolher a representação em estágios para implementar o nível 2 mas incluir uma ou duas áreas de processo de nível 3 em seu plano de melhoria. Outra possibilidade é uma organização escolher a representação contínua para guiar internamente o seu processo de melhoria e, no momento de realizar a avaliação, escolher a representação em estágios (Chrissis, 2003).

O conteúdo do CMMI pode ser classificado como "requerido", "esperado" e "informativo". O material mais importante é "requerido". Estes itens são essenciais para o modelo e para o entendimento do que é necessário para a melhoria do processo e para a demonstração de conformidade com o modelo. Em segundo lugar tem-se os itens "esperados", que embora possam não estar presentes em uma organização, por não serem essenciais, são fortes indicadores de que um item "requerido" foi alcançado. Por fim, tem-se o material informativo que constitui uma guia para a implementação do modelo. Os únicos componentes requeridos do CMMI são os objetivos. Quando um objetivo é específico de uma área de processo, é chamado de objetivo específico. Quando um objetivo pode ser aplicado em todas as áreas de processo, ele é chamado de objetivo genérico. Os componentes esperados do CMMI são as práticas. Cada prática está mapeada para apenas um objetivo. Quando uma prática é específica de uma área de processo, é chamada de prática específico. Quando uma prática pode ser aplicada em todas as áreas de processo, ela é chamada de prática genérica (Ahern, 2001, Chrissis, 2003).

Um modelo em estágios fornece um roteiro predefinido para a melhoria de processos na organização baseado em um agrupamento e ordenação de processos. O termo "estágios" vem da forma como o modelo descreve este roteiro, isto é, como uma série de estágios chamados níveis de maturidade. Cada nível de maturidade tem um conjunto de áreas de processo que indicam onde a organização deve colocar o foco de forma a melhorar o seu processo. Cada área de processo é descrita em termos das práticas que contribuem para alcançar seus objetivos. O progresso ocorre pela satisfação dos objetivos de todas as áreas de processo relacionadas a um determinado nível de maturidade. As áreas de processo do CMMI estão organizadas em quatro níveis de maturidade na representação em estágios, pois o nível 1 não possui áreas de processo.

O CMMI fornece um método de avaliação rigoroso para *benchmarking*, chamado SCAMPI, que implementa os sete princípios das avaliações CMMI (Chrissis, 2003): patrocínio da gerência senior, foco nos objetivos de negócio da organização, confidencialidade das entrevistas, uso de um método de avaliação documentado, uso de um modelo de referência de processo como base (CMMI), enfoque de equipe colaborativa, e, foco em ações para melhoria do processo.

Uma avaliação SCAMPI é dividida em três fases (Ahern, 2001): i) planejamento inicial e preparação; ii) avaliação no local; ii) relato de resultados. Uma avaliação SCAMPI deve ser liderada por avaliador treinado e autorizado pelo SEI (um *lead appraiser* SCAMPI). A equipe avaliadora deve ter no mínimo quatro membros e no máximo dez. O método de avaliação SCAMPI está em conformidade com o método de avaliação da ISO/IEC 15504 (SPICE).

4 Modelo de Referência para Melhoria de Processo de Software

O Modelo de Referência para melhoria de processo de software (MR mps) compreende níveis de maturidade e um método de avaliação (Figura 3).

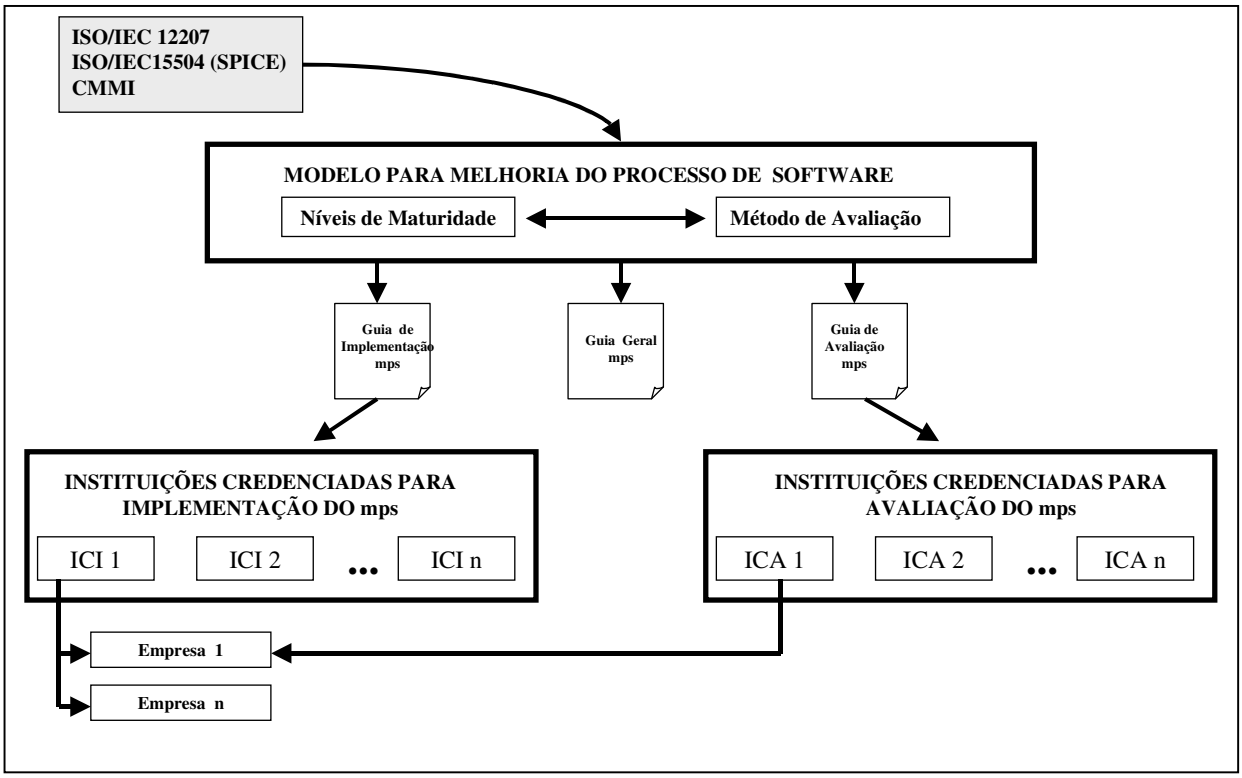


Figura 3 – Modelo de Referência para melhoria do Processo de Software (MR mps)

4.1 Implementação do MR mps

A norma de referência para os processos de ciclo de vida de software no MR mps é a ISO/IEC 12207 conforme sua atualização publicada em 2002. Esta norma pode ajudar as organizações na definição de seus processos pois ela contém uma clara definição da arquitetura, terminologia e responsabilidades inerente a processos. Essa atualização inseriu processos e acrescentou na sua descrição propósito e resultados de implementação o que possibilita a avaliação da capacidade do processo. A norma, incluindo o seu anexo, é composta por:

- Processos fundamentais: Aquisição, Fornecimento, Desenvolvimento, Operação e Manutenção.
- Processos de apoio: Documentação, Gerência de Configuração, Garantia da Qualidade, Verificação, Validação, Revisão Conjunta, Auditoria, Resolução de Problemas e Usabilidade.
- Processos organizacionais: Gerência, Infra-estrutura, Melhoria, Recursos Humanos, Gestão de Ativos, Gestão de Programa de Reuso e Engenharia de Domínio.

Cada organização interessada em implementar o MR mps deve, a partir deste conjunto, selecionar os processos que lhe são pertinentes conforme o processo de adaptação.

Desta forma, no MR mps, a norma internacional ISO/IEC 12207 é o *framework* para a definição de processos. Os resultados esperados da implementação dos processos são uma adaptação para o MRmps dos resultados esperados nos processos e atividades da ISO/IEC 12207.

A implementação do mps pode ter soluções diferenciadas dependendo das características, necessidades e desejo das organizações. Por exemplo, quando a organização desejar ter aderência de seus processos ao CMMI esta pode se apoiar nas áreas de processo deste modelo para obter diretrizes sobre como definir e implementar os seus processos. A norma ISO/IEC 12207, por sua vez, contém atividades e tarefas descritas de forma detalhada que podem auxiliar na implementação das áreas de processo. A figura 4 mostra um exemplo de como poderia ser a integração da ISO/IEC 12207 e do MR mps com o CMMI para o processo de desenvolvimento, numa organização com o objetivo de atender aos objetivos do nível 2 deste modelo.

4.2 Níveis de Maturidade

Os níveis de maturidade estabelecem uma forma de prever o desempenho futuro de uma organização com relação a uma ou mais disciplinas. Um nível de maturidade é um patamar definido de evolução de processo. Cada nível de maturidade estabelece uma parte importante do processo da organização.

No MR mps a maturidade de processo está organizada em duas dimensões: a ***dimensão capacidade*** (*capability dimension*) e a ***dimensão processo*** (*process dimension*).

A dimensão da capacidade é um conjunto de atributos de um processo que estabelece o grau de refinamento e institucionalização com que o processo é executado na organização. À medida que evolui nos níveis, um maior ganho de capacidade de desempenhar o processo é atingido pela organização. Os níveis estabelecem uma maneira racional para aprimorar a capacidade dos processos definidos no MR mps.

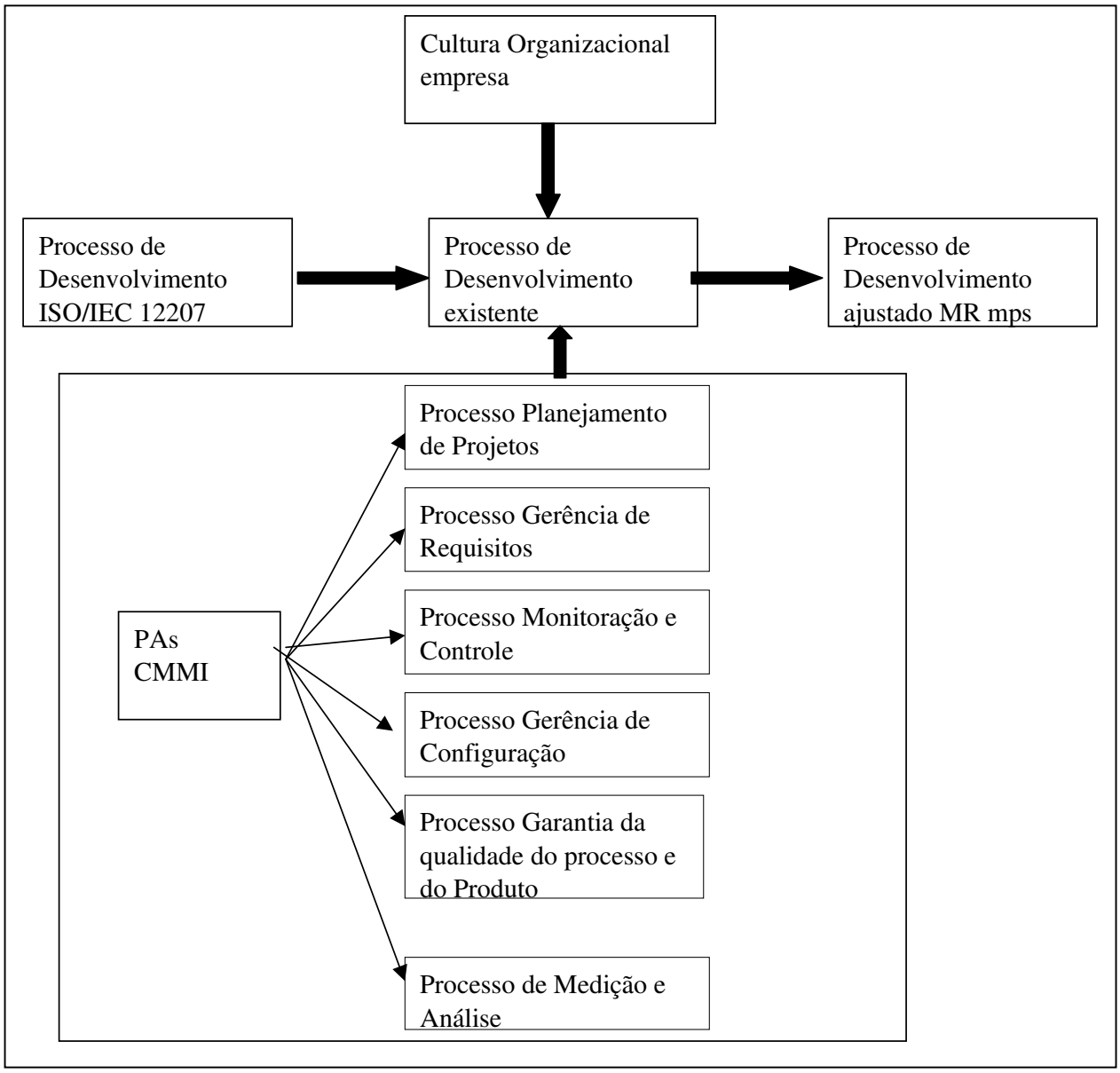


Figura 4 –Definição do Processo de Desenvolvimento baseado na ISO/IEC 12207, no MR mps e no CMMI Nível 2

A dimensão de Processos é baseada na ISO/IEC 12207 e estabelece o que a organização deveria executar para ter qualidade na produção, fornecimento, aquisição e operação de software.

A interseção dessas duas dimensões define a maturidade do processo que no MR mps são sete níveis de maturidade: A (Em Otimização), B (Gerenciado Quantitativamente), C (Definido), D (Largamente Definido), E (Parcialmente Definido), F (Gerenciado) e G (Parcialmente Gerenciado). Para cada um destes níveis de maturidade foram atribuídas áreas de processo, com base nos níveis 2, 3, 4 e 5 do CMMI em estágios. Esta divisão tem uma gradação diferente do CMMI em estágios com o objetivo de possibilitar uma implementação mais gradual e adequada

às micro, pequenas e médias empresas brasileiras. A possibilidade de se realizar avaliações considerando mais níveis permite uma visibilidade dos resultados de melhoria de processo, na empresa e no país, com prazos mais curtos. Para cada área de processo são considerados objetivos e práticas específicos, de acordo com o Nível de Maturidade em questão.

Para exemplificar a aderência da norma ISO/IEC 12207 ao CMMI mostramos na Tabela 1 o mapeamento do processo de gerência de configuração da norma para a área de processo do CMMI

Tabela 1 – Processo de Gerência de Configuração da ISO/IEC 12207 e Área de Processo Gerência de Configuração do CMMI

	ISO/IEC 12207	**CMMI**
Propósito	O propósito do processo de gerência de configuração é estabelecer e manter a integridade de todos os produtos de trabalho de um processo ou projeto e disponibilizá-los a todos os envolvidos .	A finalidade do gerenciamento da configuração de software é estabelecer e manter integridade dos produtos de software do projeto durante todo o seu ciclo de vida.
Resultados/Objetivos	Uma estratégia de gerência de configuração é desenvolvida;	As atividades de gerenciamento da configuração de software são planejadas.
	Todos os itens gerados pelo processo ou projeto são identificados, definidos e colocados sob uma linha básica (*baseline*);	Produtos de trabalho de software são identificados, controlados e disponibilizados.
	As modificações e as liberações dos itens são controladas;	Alterações identificadas nos produtos de trabalho de software são controladas
	As modificações e liberações são disponibilizadas para todos os envolvidos;	Indivíduos e grupos envolvidos com são informados sobre o *status* e conteúdo das *baselines*.
	A situação dos itens e solicitações de mudanças são registradas e relatadas;	
	A completeza e a consistência dos itens são asseguradas;	
	O armazenamento, o manuseio e a entrega dos itens são controlados.	

Consideremos novamente o exemplo anterior de organização. Para esta empresa o adequado é buscar de início uma avaliação Nível F do mps Br, cujos resultados pretendidos são compatíveis com o nível 2 do CMMI. Ao evoluir seus processos buscando o nível E do MR mps e ainda a compatibilidade com o CMMI a empresa deverá introduzir todas as áreas de processo relativas a Engenharia do nível 3 do CMMI: Desenvolvimento de Requisitos, Solução Técnica, Integração do Produto, Verificação e Validação. A evolução para o nível D implicará em implementar as

áreas de processo Treinamento Organizacional, Foco no Processo Organizacional, Definição do Processo Organizacional e Gerência Integrada do Produto. Para o nível C deverá implementar as áreas de processo Gerência de Riscos, Análise e Resolução da Decisão e Gerência Integrada de Fornecedores. Os níveis B e A correspondem, de forma idêntica, aos níveis 4 e 5 do CMMI.

4.3 Método de Avaliação

A avaliação das organizações segundo o MR mps deverá ser realizada considerando-se a aderência às áreas de processo estabelecidas para cada nível de maturidade e a adequação das práticas que implementam as áreas de processo. O método de avaliação foi definido com base na ISO/IEC 15504.

O nível de implementação das práticas relacionadas a uma área de processo é avaliada a partir de Indicadores. Estes indicadores, que devem ser definidos pela empresa para cada prática relacionada a uma área de processo, podem ser de um dos três tipos a seguir: Direto, Indireto ou Afirmação. Indicadores Diretos são produtos intermediários, resultado de uma atividade. Indicadores Indiretos são, em geral, documentos que indicam que uma atividade foi realizada. Afirmações são resultantes de entrevistas com a equipe dos projetos avaliados, onde os entrevistados relatam como uma prática foi implementada. O nível de implementação de uma prática é avaliado de acordo com quatro níveis: TI – Totalmente Implementada; LI – Largamente Implementada; PI – Parcialmente Implementada, e, NI- Não Implementada. A Tabela 2 contém as regras para caracterizar o grau de implementação das práticas, completamente aderentes à norma ISO/IEC 15504 (SPICE). Os pontos nesta escala devem ser entendidos como uma porcentagem que representa o grau de alcance. A decisão final sobre o grau de implantação de um processo é da equipe de avaliação, considerando os resultados da avaliação nos projetos avaliados.

Tabela 2 – Regras para Caracterizar o grau de implementação das práticas

Grau de Implementação da Prática	Caracterização	Grau de alcance
Totalmente Implementado	• O indicador direto está presente e julgado adequado • Existe pelo menos um indicador indireto e/ou afirmação para confirmar a implementação • Não foi notada nenhuma fraqueza substancial	> 85% a 100%
Largamente Implementado	• O indicador direto está presente e julgado adequado • Existe pelo menos um indicador indireto e/ou afirmação para confirmar a implementação • Foi notada uma ou mais fraquezas	> 50% a 85%
Parcialmente Implementado	• O indicador direto não está presente ou é julgado inadequado • Artefatos ou afirmações sugerem que alguns aspectos da prática estão implementados • Fraquezas foram documentadas	> 15% a 50%
Não Implementado	• Qualquer situação diferente das acima	0 a 15%

Uma empresa é considerada de nível A, B, C, D, E, F ou G se todas as suas áreas, unidades, divisões ou setores tiverem sido avaliados como naquele nível. Uma empresa, entretanto, pode desejar ter avaliado apenas um ou alguns de seus setores, áreas, unidades ou divisões (organização a ser avaliada). É possível que, como resultado de uma ou mais avaliações, partes

de uma empresa tenham alcançado um determinado nível e partes da mesma um outro nível. Em qualquer caso, o documento comprobatório da avaliação deverá explicitar o que foi objeto de avaliação (escopo da avaliação) e o nível resultante de maturidade..

Para realização de uma avaliação devem ser submetidos todos os projetos concluídos e todos os projetos em andamento a partir da implementação MR mps na empresa ou na organização que será avaliada. Durante o planejamento da avaliação, a instituição avaliadora deve selecionar um subconjunto suficiente de projetos que garanta a representatividade da organização a ser avaliada. Este número, entretanto, não deve ser inferior a dois projetos concluídos e dois projetos em andamento. Algumas empresas podem desenvolver um único produto. Isto entretanto não é impedimento para a avaliação, pois projetos são entendidos em sentido amplo, incluindo projetos de manutenção no produto. O resultado de uma avaliação tem validade de dois anos.

Uma Guia Geral descreve o MR mps. Diretrizes para implementação, avaliação e aquisição segundo o MR mps são descritas em guias específicas. Essas guias estão sendo elaboradas e refinadas pela equipe técnica do modelo. Tem-se neste momento uma versão preliminar da Guia Geral, orientando a experiência-piloto. A partir das guias específicas, diferentes instituições poderão definir sua estratégia de implementação e/ou avaliação de acordo com o MR mps e submetê-las para credenciamento junto ao Fórum de Credenciamento e Controle (FCC), formado por representantes do governo, da Sociedade SOFTEX e das Universidades. Após o credenciamento pelo FCC, uma instituição está apta para apoiar empresas na implementação do MR mps e/ou avaliar a aderência das mesmas ao modelo. Para solicitar o seu credenciamento, as instituições devem submeter previamente um documento com o seguinte conteúdo:

- Apresentação da Instituição proponente, contendo dados da organização com ênfase na experiência em processos de software
- Estratégia de implementação do modelo de referência, caso deseje se credenciar para isto
- Estratégia de avaliação segundo o método de avaliação, caso deseje se credenciar para isto
- Estratégia para seleção e treinamento de consultores de implementação, se pertinente
- Estratégia para seleção e treinamento de avaliadores, se pertinente
- Se pertinente, lista de consultores de implementação, onde para cada candidato a consultor deve ser apresentado o *curriculum vitae*, o comprovante de presença em curso introdutório sobre o MR mps e comprovante de aprovação em prova de conhecimentos sobre o modelo .
- Se pertinente, lista de avaliadores, onde para cada candidato a avaliador deve ser apresentado o *curriculum vitae*, o comprovante de presença em curso introdutório sobre o MR mps e o comprovante de aprovação no curso sobre o método de avaliação.

5 Experiência-piloto e Próximos Passos do Projeto

A partir de dezembro de 2003, inicialmente, foram identificados os seus objetivos estratégicos, estabelecido o Plano de Ação e definido o MR mps. A seguir, foram realizadas experiências em empresas, sob a coordenação do grupo responsável pela definição do modelo, aprimorando o MR mps de modo iterativo.

No Rio de Janeiro, o MR mps está sendo implementado pela COPPE/UFRJ em 18 pequenas e médias empresas, que constituem dois grupos organizados pela RIOSOFT. Estas empresas partilharam as atividades de treinamento, que constaram de 44 horas de aula em temas de Engenharia de Software e 20 horas no MR mps e nos processos a serem implementados. Foram definidas duas estratégias de implementação. Algumas empresas optaram por iniciar seu processo

de melhoria seguindo rigorosamente os níveis do MR e, desta forma, estão concentradas nas áreas de processo do nível de maturidade G. Outro conjunto de empresas decidiu iniciar o trabalho englobando os níveis F e G e a área de processo Medição e Análise, de forma a já iniciar o processo de melhoria com a implementação das áreas de processo equivalentes ao nível de maturidade 2 do CMMI. As duas estratégias são perfeitamente compatíveis com o MR e com os objetivos do projeto mps. Para apoiar a implementação do modelo estas empresas contam com consultores da COPPE/UFRJ e um ambiente de desenvolvimento de software com ferramentas de apoio desenvolvidas para apoiar as áreas de processo (Oliveira, 2004, Villela, 2004, Montoni, 2004, Farias, 2003).

Outros grupos estão se formando em diferentes locais do país e iniciarão suas atividades no 2º semestre de 2004. Teve, também, início a implementação do MR mps em três grandes organizações do governo brasileiro, o que está sendo feito segundo o Modelo de Negócio Específico.

Foram, também, iniciadas as atividades de treinamento e credenciamento. Foram realizados três *workshops* do Projeto mps (em São Paulo, Brasília e Recife) e um curso introdutório ao modelo (no Rio de Janeiro), com ampla participação de empresas, governo e pesquisadores. Já foi, também, realizada a primeira prova de conhecimento para implementadores do modelo. A partir dos resultados da prova se dará início ao credenciamento de instituições para implementação, pelo Forum de Credenciamento e Controle (FCC). Espera-se poder credenciar instituições para implementação em várias cidades do Brasil, entre elas São Paulo, Campinas, Recife, Rio de Janeiro, Fortaleza, Porto Alegre, Brasília, Belo Horizonte e Curitiba. Outros cursos e *workshops* sobre o modelo estão agendados em Porto Alegre, Campinas, Manaus, Fortaleza, Rio de Janeiro e Belo Horizonte. Serão realizadas duas outras provas de conhecimento para implementadores do modelo ainda este ano em outubro e dezembro. Também, esperamos dar início ao credenciamento de instituições avaliadoras, de forma que as primeiras empresas possam ser avaliadas no início de 2005.

6 Conclusão

Neste artigo apresentamos o projeto mps. Este projeto vem alcançando um alto grau de adesão por parte de empresas privadas e organismos governamentais. A busca por uma solução que realmente atenda à realidade brasileira tem envolvido um amplo debate e o esforço conjunto de uma grande equipe, com representantes de várias regiões do país.

Algumas lições aprendidas já podem ser relatadas a partir deste esforço e das primeiras experiências de implementação do modelo: (i) é necessário ter-se um modelo abrangente que permita uma grande variedade de formas de implementação, dependendo das particularidades e do porte das empresas envolvidas; (ii) o Modelo de Negócio Cooperado tem-se mostrado adequado e capaz de atender à realidade de pequenas e médias empresas por permitir a implementação do modelo a um custo mais acessível; (iii) no Modelo de Negócio Cooperado, um aspecto fundamental para o sucesso da implementação do MR mps nas experiências-piloto tem sido a experiência e grau de formação dos implementadores, bem como a existência de uma coordenação do grupo de empresas que direcione adequadamente as ações do grupo; (iv) para grandes empresas, empresas com grande grau de especificidade ou para empresas que já tenham um processo implementado, o Modelo de Negócio Específico e personalizado tem-se mostrado mais adequado; (v) o grau de exigência para credenciamento de instituições implementadoras e

avaliadoras tem sido valorizado por todos os segmentos envolvidos; (vi) as atividades de formação têm sido avaliadas muito positivamente e pretendemos aumentar sua abrangência e profundidade.

O projeto tem sete diferenciais que o caracterizam: (i) sete níveis de maturidade que permitem uma implementação gradual, adequada à micro, pequena e média empresa, e que permitem aumentar a visibilidade do processo de melhoria; (ii) compatibilidade com a ISO/IEC 12207, a ISO/IEC 155504 (SPICE) e CMMI; (iii) ser criado para a realidade brasileira; (iv) custo acessível; (v) avaliação periódica (de 2 em 2 anos); (vi) grande potencial de replicabilidade no Brasil e em outros países; e, (vii) ter sido definido e ser implementado em forte interação universidade-empresa, o que constitui um catalizador do desenvolvimento tecnológico e de negócios.

Referências

(Ahern, 2001) Ahern,D.M., Clouse,A, Turner,R. CMMI Distilled: a Practical Introduction to Integrated Process Improvement, Addison-Wesley, 2001

(Chrissis, 2003) Chrissis,M.B., Konrad,M, Shrum,S. CMMI: Guidelines for Process Integration and Product Improvement, Addison-Wesley, 2003

(Farias, 2003) Farias,L.L., Travassos,G,H., Rocha, AR Managing Organizational Risk Knowledge, Journal of Universal Computer Science, vol 9 n 7 (2003), 670- 681, julho 2003

(ISO, 2000) ISO 9001:2000 - Sistemas de Gestão da Qualidade – Requisitos, 2000

(ISO/IEC PDAM 12207, 2002) INTERNATIONAL STANDARD ORGANIZATION. ***ISO/IEC 12207 Information Technology - Amendment to ISO/IEC 12207***. Montreal: ISO/IEC JTC1 SC7, 2002.

(ISO/IEC, 2003) ISO/IEC 15504 –1 Information Technology – Process Assessment, - Part 1: Concepts and Vocabulary, 2003

(Montoni, 2004) Montoni,M., Miranda, R., Rocha,A.R., Travassos,G. "Knowledge acquisition and Communities of Practice: an Approach to Convert Individual Knowledge to Multi-Organizational Knowledge", VI International Workshop on Learning Software Organizations. Banff, Canada, junho 2004

(NBR ISO/IEC 12207, 1998) ABNT - ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **ISO/IEC 12207 - Tecnologia de Informação - Processos de ciclo de vida de software.** Rio de Janeiro: ABNT, 1998.

(Oliveira, 2004) Oliveira, K,, Zlot,F., Rocha, AR., Travassos,G., Galotta,C., Menezes,C. Domain Oriented Software Development Environment, Journal of Systems and Software, vol 72/2 pp 145-161

(Salviano, 2001) Salviano,C., Cunha,M.A.V.C., Côrtes,M.L, Oliveira,W.L. SPICE in Rocha,A.R.C., Maldonado,J.C, Weber, K.C. (eds) Qualidade de Software: Teoria e Prática. São Paulo, Prentice Hall, 2001

(Veloso et al, 2003) Veloso, F., Botelho, A. J. J., Tschang, T., Amsden, A. Slicing the Knowledge-based Economy in Brazil, China and India: A Tale of 3 Software Industries. Report. Massachussetts Institute of Technology (MIT), setembro 2003.

(Villela, 2004) Villela,K., Santos,G., Schnaider,L, Rocha,A.R., Travassos,G. "Building ontology based tools for a software development environment", VI International Workshop on Learning Software Organizations. Banff, Canada, junho 2004

(Weber, 1995) Weber, K. C., Pinheiro, M. Software Quality in Brazil. Quality World Magazine, vol. 21, issue 1.1. The Institute of Quality Assurance (IQA). London, UK, novembro l995.

(Weber, 1997) Weber, K.C., Almeida, R.A.R., Amaral, H.G., Gunther, P.S., Xavier, J.H.F., Loures, R. "ISO 9001/TickIT Certification in Brazilian Software Companies". 5th International Conference on Software Quality Management (SQM'97). Bath, UK, março 1997.

(Weber, 2001) Weber, K. C., Rocha, A. R. C., Nascimento, C. J. Qualidade e Produtividade em Software, 4ª edição renovada. São Paulo, Makron Books, 2001.

Experimenting With the TPC-W E-commerce Benchmark

Mehdi Khouja, Farouk Kamoun
Université de la Manouba, Ecole Nationale des Sciences de l'Informatique (ENSI),
La Manouba, Tunisia, 2010
mehdi.khouja@cristal.rnu.tn, frk.kamoun@planet.tn

Catalina M. Lladó, Ramon Puigjaner
Universitat de les Illes Balears, Departament de Matemàtiques i Informàtica,
Palma de Mallorca, Spain, 07071
cllado@uib.es, puxi@uib.es

Abstract

The success of an e-commerce site highly depends on its performance characteristics. These, are very difficult to asses for the given software and hardware characteristics of the specific system. The TPC-W is a benchmark aimed at evaluating e-commerce sites. This paper presents an implementation of this benchmark and the experimentation process that has been carried out in order to evaluate it. A full factorial experimental design has been used with the factors, number of emulated browsers, their profile and the number of processors in the server machine. The analysis of the results is done in terms of the TPC-W main metric, Web Interaction Per Second (WIPS) and it shows the effect of the variation of the factors above mentioned on the TPC-W throughput.

Keywords: TPC-W, Benchmarking, E-commerce, Computer Evaluation.

1 INTRODUCTION

Nowadays, the Internet is deeply affecting the commerce. However, e-commerce sites are at a risk of being overwhelmed by large numbers of customers and therefore offering them a very bad performance. Hardware and software vendors and commerce service providers need to know the best configuration to use in order to fit the customers demand. One possibility for the sizing and capacity planning of computer systems is the use of benchmarks. TPC-W (specified by the Transaction Processing Performance Council), is a benchmark aimed at evaluating sites that support e-commerce activities. The business model [4] of this benchmark is an e-commerce web site that sells products over the Internet. The site provides e-business functions that allow customers to browse through selected products (e.g., best sellers or new products), search information on existing products, see product detail, place an order, or check the status of a previous order.

The benchmark results are highly depending on the benchmark specification, workload, and design and implementation of the benchmark application. Therefore experimentation with the benchmark needs to be done in order to evaluate this dependency. Clearly, the first step is to design and develop a TPC-W benchmark application, taking into account that the results can vary between different application implementations. This paper presents a TPC-W benchmark implementation that has been developed at the Universitat de les Illes Balears and the experimental design process done in order to evaluate the utility of this benchmark.

Related work can be found in [1], where a TPC-W implementation and a set of experiments conducted to study how the response time varies as the request arrival rate increases are presented. Another TPC-W implementation and its evaluation is presented in [2]. Nevertheless, those implementations are different from the one presented in this work and the experiments they perform also differ substantially.

The paper is structured as follows: while section 2 describes the TPC-W specification, section 3 shows the main characteristics of the specific implementation of the TPC-W and the system configuration that has been used to carry out the experimentation process. Following, section 4 covers the experimental methodology and its application to the case under study and section 5 analysis and discusses the experimental results obtained. Finally, conclusions and future work are covered in section 6.

2 TPC-W OVERVIEW

TPC-W is a transactional Web benchmark with a client-server architecture which represents a typical e-commere environment. It covers both, the server application and the workload generator. The e-commerce environment is characterized by multiple on-line browsing sessions, static and dynamic web pages services and accesses and updates to a database [6]. The site maintains a catalog of items that can be searched by a customer. TPC-W specifies that the site maintains a database with information about customers, items in the catalog, orders, and credit card transactions.

Three components constitute the TPC-W system: a set of Emulated Browsers (EB) which is called RBE (Remote Browser Emulator), a system under test (SUT), which mainly includes a web and a database server, and a Payment Gateway Emulator (PGE). Figure 1 shows the TPC-W environment.

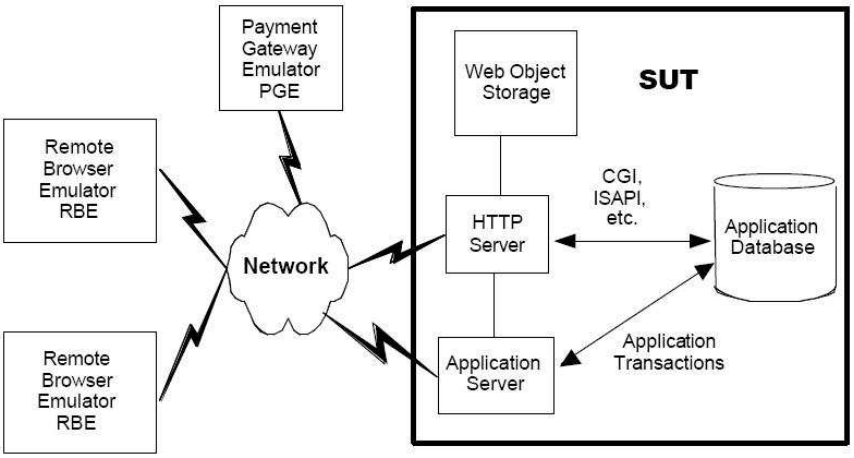


Figure 1: TPC-W environment

Following, the server application (SUT) and the workload generator (RBE) are described in more detail, as well as the performance metrics that the benchmark reports.

2.1 The Workload Generator

The emulated browsers drive the TPC-W workload. They emulate human customers sending and receiving HTML contents using HTTP and TCP/IP over a network connection. The EBs can browse, search the site's bookstore, view product details, add products to the shopping cart and buy added products. The customer activities are categorized into 14 specific web interactions, though TPC-W also classifies those into two broad categories:

- *Browse* interactions involve browsing and searching but no product ordering activity.
- *Order* interactions involve product ordering activities only.

Each of the 14 web interactions are characterized by the number of images included in the response page, the number of database table joints that require, and a maximum response time. In order to consider an interaction as valid, it needs to be executed within this maximum value. Table 1 shows these characteristics for each TPC-W web interaction.

On the client site of the benchmark, each emulated customer submits a series of requests within a user session. TPC-W defines the minimum user session duration (USMD), which is generated at the starting of a session from a exponentially distributed function with mean equal to 15 minutes. The user session duration is defined as the time elapsed between the first request executed by the EB and the current time. Therefore, when the user session duration reaches or passes the USMD, the session finishes. At the end of a session the EBs must close all the TCP/IP and SSL (Secure Sockets Layer) connections and restart a new session with the generation of a new USMD.

The EBs wait a period of time between two consecutive requests in order to simulate the user think time (TT). This time is generated using a negative exponential distribution function with mean equal to 7 seconds.

A user session is characterized by a browsing to ordering interaction ratio. TPC-W specifies three session types:

Table 1: Characteristics of TPC-W Web Interactions

Name	Dynamic Generation	Number of Joints	Number of Images	Max Response Time (s)	Interaction Type
Admin Confirm	Yes	4	5	20	Order
Admin Request	Yes	2	6	3	Order
Best Seller	Yes	3	9	5	Browse
Buy Confirm	Yes	1	2	5	Order
Buy Request	Yes	1	3	3	Order
Customer Registration	No	N/A	4	3	Order
Home	Yes	1	9	3	Browse
New Product	Yes	2	9	5	Browse
Order Display	Yes	1	2	3	Order
Order Inquiry	No	N/A	3	3	Order
Product Detail	Yes	2	6	3	Browse
Search Request	No	N/A	9	3	Browse
Search Result	Yes	2	9	10	Browse
Shopping Cart	Yes	1	9	3	Order

- Browsing mix: 95% of browsing and 5% of ordering
- Shopping mix: 80% of browsing and 20% of ordering
- Ordering mix: 50% of browsing and 50% of ordering

The customer behaviour model graph (CBMG), shown in Figure 2, specifies the navigational behaviour of the EBs through the web site. Each interaction represents a state in the graph. The transition from a state to an another is characterized by a probability of transition that depends on the EB session type. For instance, the Shopping Cart state represents a state in which items can be added or deleted from the shopping cart and customers have to go through the Customer Registration state before they can reach the Buy Request (customers need to register with the site before placing an order).

2.2 The System Under Test

The System Under Test (SUT) comprises all the components which are part of the application being emulated. This includes network connections, Web servers, application servers and database servers. It does not include the RBE or the PGE [6]. The two main components of the SUT are the database and the web server.

The TPC-W database consists of 8 tables. These tables store users, authors, books, orders and credit cards information. The database size depends on two scalability parameters: the number of emulated clients and the number of bookstore items. The second parameter must be chosen from a list of five values specified by TPC-W: 1K, 10K, 100K, 1M and 10M. Database scalability rules (see [3]) are shown in Figure 3. Database interactions must verify the ACID properties: Atomicity, Consistence, Isolation and Durability.

When the SUT needs to finalize an order interaction, it uses a third component which is the PGE. The PGE generates an authentification identifier and then the buying transaction is accomplished. The connections between the SUT and the PGE are over the secure socket layer.

2.3 Performance Metrics

TPC-W has two types of performance metrics: a throughput metric and a cost to throughput ratio metric. There are three throughput metrics depending of the type of session. The main one measures the average number of valid Web Interactions executed Per Second during a simulation interval in which all the sessions are of the shopping type (WIPS). As said above, valid interactions are those which the response time is under the maximum response time specified by TPC-W (see Table 1). There are also two secondary throughput metrics corresponding to the other TPC-W mixes: the WIPb, measures the average number of valid Web Interactions completed during an interval in which all sessions are of the browsing type. The other, called WIPSo, measures the average number of valid Web Interactions per second completed during an interval in which all sessions are of the ordering type.

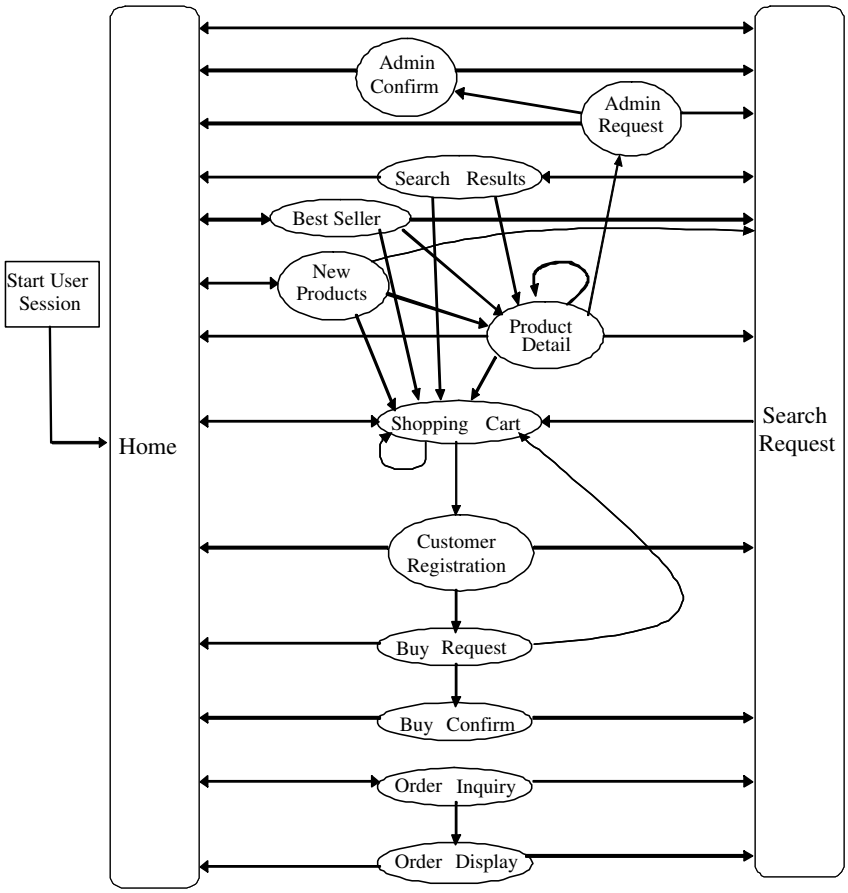


Figure 2: TPC-W Customer Behaviour Model Graph

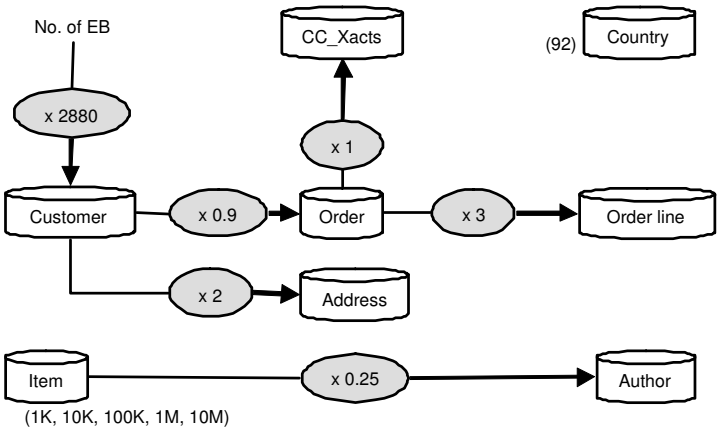


Figure 3: Database Scalability Rules

The TPC-W specification states that in order to consider the value reported for the WIPS as valid, it must satisfy the following bounding conditions: the upper bound corresponds to the WIPS that would be reported for a response time equal to zero and the lower bound is set as 50% of the upper bound. Since the user think time has a mean value of 7 seconds, the upper bound is $EB/7$ and the lower bound is $EB/14$.

The cost related metric specified by TPC-W is $\$/WIPS$ and indicates the ratio between the total cost of the system under test and the number of WIPS measured during a shopping interval. Total cost includes purchase and maintenance cost for all hardware and software components.

Other performance metrics that can be reported are CPU and memory utilization, database I/O activities, etc.

3 TPC-W IMPLEMENTATION AND SYSTEM CONFIGURATION

The evaluated implementation of the TPC-W has been developed in Java. Figure 4 shows the specific architecture used in this implementation.

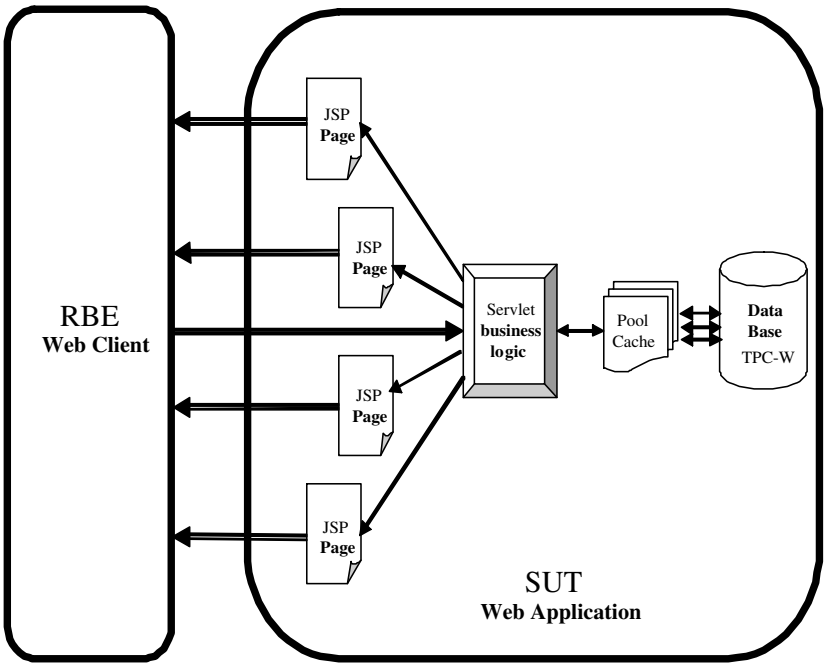


Figure 4: TPC-W Architecture

The RBE component has a graphical interface allowing the experimenter to introduce the following execution parameters:

- Web Server IP address.
- Number of emulated clients.
- Type of mix.
- Simulation time.

The TPC-W web site is implemented using Java Server Page (JSP). The web site is based on the servlet-centric approach. Client requests arrive to the central servlet, which takes charge of processing them, retrieving the required information from the data base through the PoolMan (an embeddable object pooling and caching library) and, once done all required actions, routing the response information to the JSP front-end pages. The access to the database is done through JDBC (Java Database Connectivity).

For the experimentation process, three different computers are used, one for each component of TPC-W (RBE, SUT and PGE). Table 2 shows the SUT configuration in the experimentation. The number of bookstore items in the database is 1000.

Table 2: SUT Configuration

Component	Description
Operating System	Microsoft Windows 2000 Server
CPU	2 x Intel Pentium XEON 2.4 GHZ
RAM	2 GB
HTTP Server	Apache 2.0.44
Servlet Engine	Tomcat 3.3.1
Java Virtual Machine	Java 2 Runtime Environment, Standard Edition 1.3.1
Database	Oracle 9i Enterprise Edition Release 9.2

4 EXPERIMENTAL DESIGN FORMULATION

Once the measurement tool is developed and the physical configuration established, the logical configuration of the measurement environment has to be specified. Next, the measurement phase has to be carried out and the reliability of the results has to be tested. This section describes the workload characteristics, as well as the subsequent experimental design formulation.

The measure under study is the main throughput metric, WIPS. A full-factor experimental design is used since there are three factors which influence we are intending to study, namely, *Number of Processors*, *Number of Emulated Clients*, *Emulated Clients Profile*. Table 3 shows the different levels for these factors, making a $2 * 14 * 3$ factorial design.

Table 3: Design Factors and Levels

Factor	Levels
Number of Processor	2 levels : 1, 2
Number of Emulated Clients	14 levels : 10, 20, 30, ..., 130
Emulated Clients Profile	3 levels : Shopping mix, Browsing mix, Ordering mix

Further, to isolate the possible effects due to the instability of the real system where the benchmarks are executed, more than one measure for each experiment (where each combination of the different levels of factors is an experiment) is obtained. In this study, five measures (or replications) for each experiment have been used.

5 EXPERIMENTAL RESULTS

The full factorial design with replications is used because it identifies the influence of each of the factors as well as the interaction between these factors [5]. Additionally, the replications allow for the isolation of the influence of experimental errors.

The first step in the experimental design process is the analysis of variance (ANOVA). This analysis revealed that the influence of the number of EBs, their profile and their interaction, on the throughput value obtained (WIPS) is statistically significant. As shown in Table 4, ANOVA allocates the total variation of WIPS due to the type of clients as 42.15% (SS_M/SS_T), due to the number of EBs as 27.43% (SS_B/SS_T), and due to the interactions between them as 25.38% (SS_{MB}/SS_T). However, the effect due to the variation of the number of processors is 0.17% (SS_P/SS_T).

5.1 The Type of Mix Effect

Figure 5 shows the variation of the WIPS depending on the number of EBs, for the three different types of mixes when the SUT is running with one processor. Similarly, figure 6 shows the variation of the WIPS depending on the number of EBs, for the three different types of mixes when the SUT is running with two processors.

As shown in the above mentioned graphs, the WIPS increases more or less linear as the number of EBs increases until it reaches a maximum value from which the WIPS drop, due to the saturation of the system. The number of EBs in the system when this maximum is reached it varies substantially depending on the type of mix we are looking at. For instance, when executing the SUT with one processor, the maximum occurs when there are 90 clients for the *shopping* mix, 110 clients for the *browsing* mix and 30 clients for the *ordering* mix. Similar results are shown when executing the SUT with 2 processors.

Table 4: Analysis of Variance for the throughput

Source of Variation	Sum of Squares	Degrees of Freedom	Mean Square	F	P-value
Processor	$SS_P = 10,46$	1	10,46	13,94	0,0002
Mix	$SS_M = 2557,62$	2	1278,81	1704,29	0,00
EB	$SS_B = 1664,89$	12	138,74	184,90	0,00
Processor∗Mix	$SS_{PM} = 16,19$	2	8,098	10,79	0,00
Processor∗EB	$SS_{PB} = 11,98$	12	0,99	1,33	0,19
Mix∗EB	$SS_{MB} = 1540,01$	24	64,167	85,51	0,00
Processor∗Mix∗EB	$SS_{PMB} = 32,41$	24	1,35	1,8	0,013
Error	$SS_E = 234,10$	312	0,75		
Total	$SS_T = 6067,72$	389			

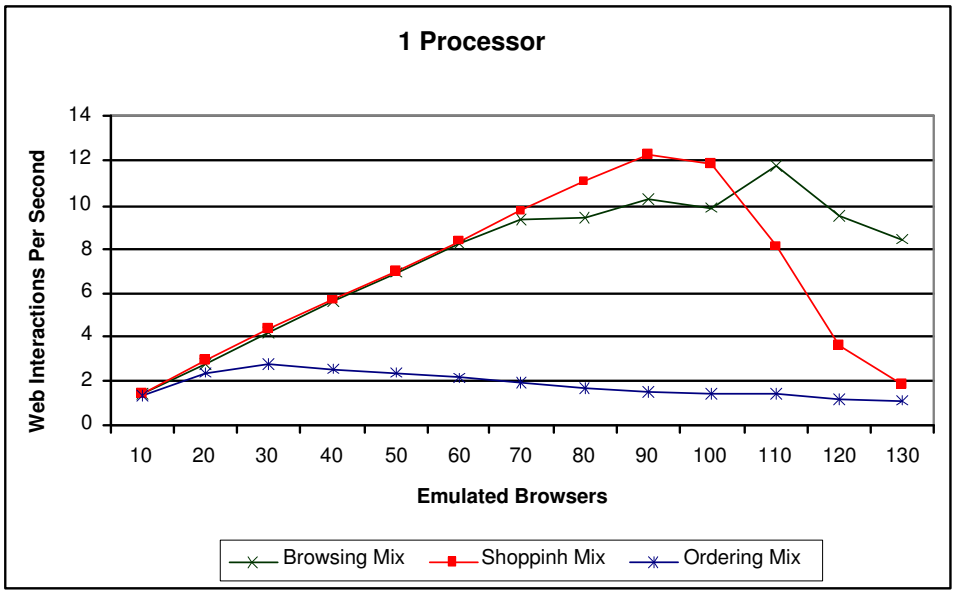


Figure 5: WIPS for 1 Processor

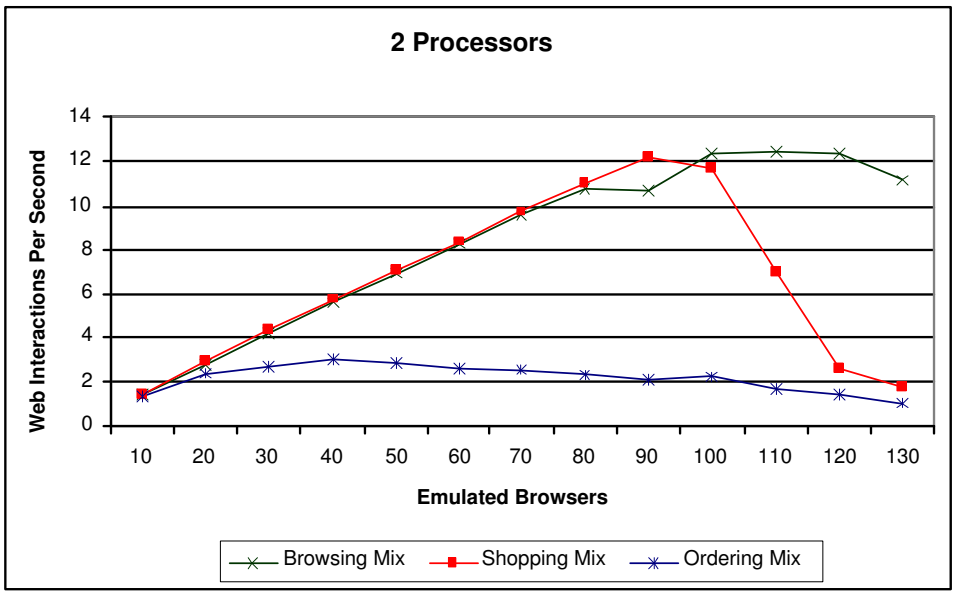


Figure 6: WIPS for 2 Processors

Therefore, it is shown that there is a significant difference between the *shopping*, *browsing* and *ordering* mixes. In fact the *ordering* mix (which leads to the server saturation very much earlier than the other mixes) induces more interaction between the SUT and the PGE for the order transaction authentification. This step requires the use of SSL connections which have a much higher response time when compared with non SSL connections. Being the WIPS the inverse of the response time, the site throughput is much lower when executing the *ordering* mix than when executing the other two mixes. Comparing the *browsing* and the *shopping* mix, the difference is not as substantial. Even though the *shopping* mix involves 15% more ordering interactions than the *browsing* one, the results do not show such a difference in terms of the throughput. Clearly, not only the use of SSL connections influences the throughput results.

5.2 The Processor Effect

The number of processors in a web server is a key factor for e-commerce server scalability. In order to evaluate the processor influence on WIPS, the benchmark has been executed using one and two processors. Figures 7, 8 and 9 show the results obtained for the *browsing*, *ordering* and *shopping* mix respectively when the number of EBs increases from 10 to 130.

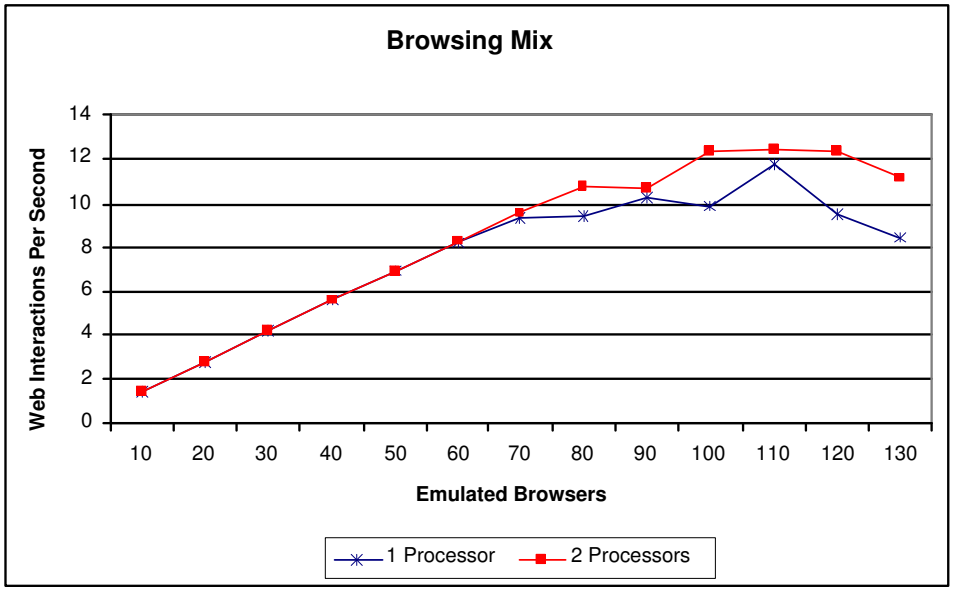


Figure 7: Browsing Mix

The 3 graphs have two well differentiated phases, the initial one, where the WIPS increase linearly as the number of EBs do, in which the throughput is equal to the best one that can be obtained (EB/7). For the *browsing* and *ordering* mix this initial phase is a bit longer in the 2-processors case than in the 1-processor case, though the difference is very subtle. The *shopping* mix results are even more disappointing since the 1-processor and 2-processors results are exactly the same for this initial phase.

Looking at the second phase of the graphs for the browsing and ordering mix, a higher number of WIPS can be observed in the 2-processors case, as a natural follow up of the first phase. On the contrary, the *shopping* mix results for this second phase show even better WIPS for the 1-processor case than for the 2-processor case. This could be due to the overhead introduced when using 2-processors since they need to communicate with each other. Very recently, we also learned that even though Oracle detects automatically the number of CPUs in the system, for Oracle databases to be concurrently and efficiently accessed by more than one processor, they have to be instantiated specifically. We have done some preliminary experiments using this sort of instantiation and also have added some *hints* to the data base queries for a more efficient use of the 2-processors. Unfortunately, the results obtained are not any better than the previous ones. More experiments will be carried out in the near future to confirm these first set of results.

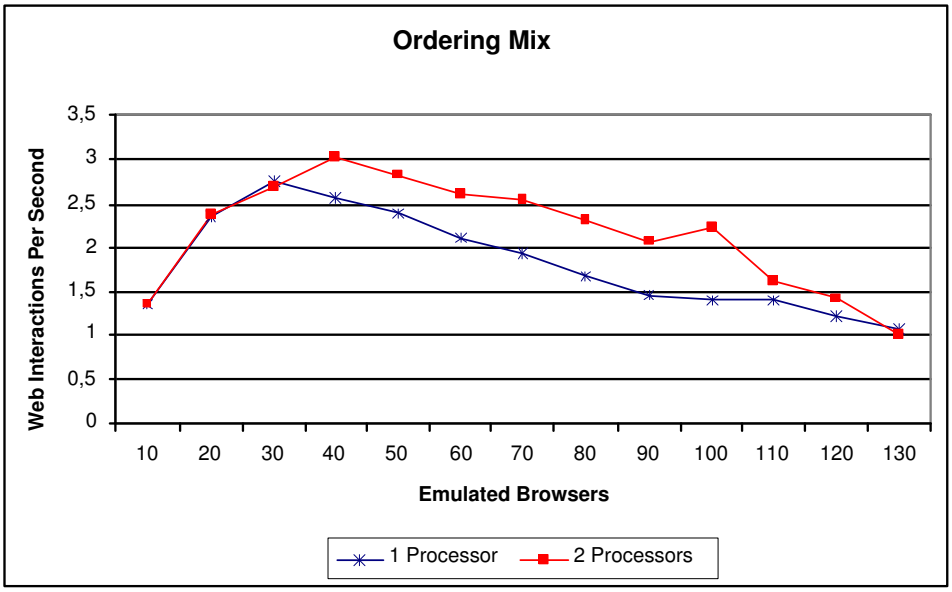


Figure 8: Ordering Mix

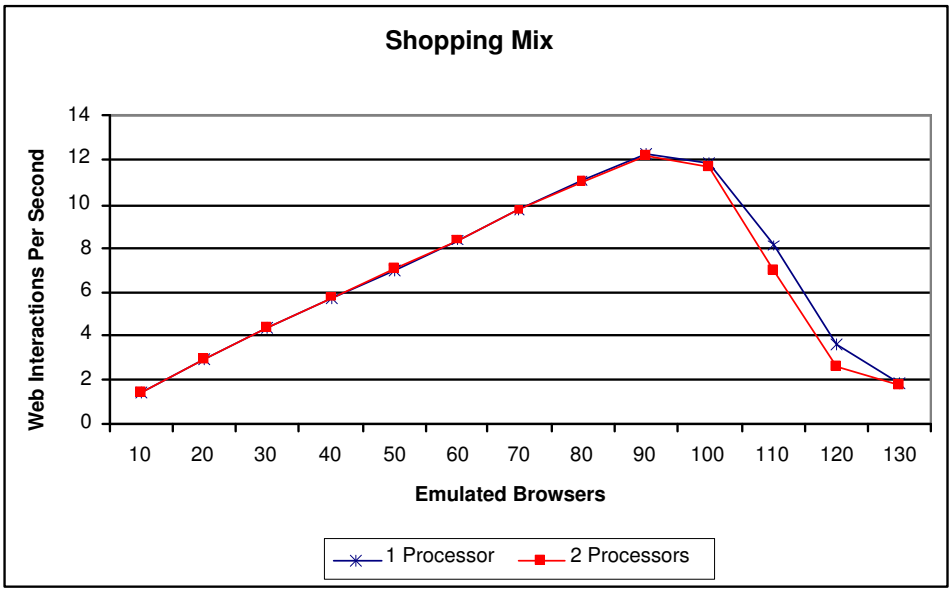


Figure 9: Shopping Mix

6 CONCLUSIONS AND FUTURE WORK

Due to the fast growing of the Internet, web servers are faced to the increasing number of fast, secure and highly available service requirements. Choosing the hardware and software configuration of an e-commerce site is not an easy task to do. Benchmarking techniques can be used to compare possible alternatives and the TPC-W is a benchmark oriented to e-commerce environments. We have implemented this benchmark and carried out a set of experiments in order to test its power to help the evaluation of e-commerce sites configurations. The analysis of the results show that the influence on the throughput results of the number of EBs, their profile and their interaction, is statistically significant. However, influence due to the variation of the number of processors is minimal. It is difficult to asses the scalability of the benchmark, though it seems that the CPU is not the bottleneck of the TPC-W implementation tested. In this sense, the bottleneck needs to be found and the benchmark implementation modified in order to remove this bottleneck. Future work also includes a more detailed study of the scalability, running the benchmark in a multiprocessor machine (with as much as 16 processors). Moreover, this implementation will be also compared with the one presented in [2], running both implementations on exactly the same hardware configuration in order to compare the influence of the software design and implementation on the throughput results.

References

[1] R. Dodge D. A. Menasce and D. Barbara. Testing e-commerce site scalability with tpc-w. *Computer Measurement Group Conference*, December 2-7 2001.

[2] Daniel F. Garcia and Javier Garcia. Tpc-w e-commerce benchmark evaluation. *IEEE Internet Computing*, February 2003.

[3] Daniel. A. Menasce. TPC-W: A Benchmark for E-commerce. *IEEE Internet Computing*, May/June 2002.

[4] Daniel A. Menasce and Virgilio A.F. Almeida. *Scaling for E-Business*. Prentice Hall Inc., 2000.

[5] D. C. Montgomery. *Design and Analysis of Experiments*. John Wiley and Sons, Inc., N.Y, 5th edition, 2001.

[6] The Transaction Performance Processsing Council (TPC). TPC Benchmark W Specification Version 1.8, February 2002.

Estruturação de Descrições de Casos de Uso através de Mecanismos de Extensibilidade da UML

Gabriel Silva Bornia, BSc.
Roberto Tom Price, Eng., MSc., D. Phil.

Universidade Federal do Rio Grande do Sul – UFRGS
Instituto de Informática
Av. Bento Gonçalves, 9500
Porto Alegre – RS – Brasil
e-mail: {bornia, tomprice}@inf.ufrgs.br

Abstract

This paper presents a structured representation of use case descriptions using stereotyped activity diagrams. The use of the extensibility mechanism of UML is used to configure language elements to be used for the description of system behavior. A way of representing use case descriptions in different levels of abstraction is shown, and ways of associating between descriptive elements and the static model of the system. A CASE tool is presented to demostrate the proposed use case description method.

Key-words: Use case, use case description, collaboration case, UML, activity diagram, extensibility mechanism, CASE tool.

Resumo

Este trabalho apresenta uma forma de representação estruturada de descrições de casos de uso através do uso de diagramas de atividade estereotipados. O uso da extensibilidade da UML permite configurar elementos da linguagem de tal forma que esta possa também ser utilizada para a descrição do comportamento do sistema. É apresentada uma forma de representação de descrições de casos de uso em vários níveis de abstração, bem como a associação entre elementos da descrição e o modelo estático do sistema. Uma ferramenta CASE é apresentada como prova de conceito para o método de descrição proposto.

Palavras-chave: Caso de uso, descrição de casos de uso, casos de colaboração, UML, diagramas de atividade, mecanismos de extensibilidade, ferramenta CASE.

1 Introdução

Na análise de sistemas orientada a objetos a modelagem de casos de uso possui um papel fundamental. Os casos de uso são o principal mecanismo de levantamento de requisitos [24], caracterizando-se por uma representação narrativa do comportamento do sistema [14][15].

As descrições de casos de uso caracterizam-se por serem descrições do funcionamento do sistema sem, no entanto, entrar em detalhes específicos de como o mesmo será implementado. Diversos autores [6][8][10] defendem que os casos de uso devem ser o mais simples possível, uma vez que devem ser entendidos pelos usuários do sistema – que os utilizam como forma de validação dos requisitos levantados pelo analista – e servir como base aos demais dos envolvidos na construção do sistema.

A necessidade de se criar descrições simples que possam ser entendidas por usuários comuns representa um obstáculo ao uso de mecanismo mais formais de descrição [9] que possam ser utilizados de forma estruturada por projetistas, desenvolvedores e gerentes de projeto.

Existem diversas formas de se representar descrições de casos de uso. A mais comum delas é a forma textual, que embora possua diversos estilos propostos [1][3][17] continua praticamente livre de uma estrutura que permita a modelagem da descrição. Existem diversas propostas de formalizações de casos de uso [12][13] de difícil disseminação no mercado. UML é uma linguagem de modelagem [4] amplamente difundida. Geralmente, os casos de uso são modelados com esta linguagem, mas a interação entre o sistema e o ator (descrição do caso de uso) é realizada de forma narrativa [21]. As descrições dessas interações podem ser realizadas através de diversos elementos da linguagem UML como, por exemplo, diagramas de atividade [3], diagramas de seqüência, diagramas de estado, entre outros. UML possui a facilidade de estender os seus elementos através do conceito de estereótipos, o que poderia ser utilizado para ajudar a enriquecer uma descrição feita com os elementos desta linguagem.

O objetivo deste trabalho é a representação de descrições de casos de uso através de mecanismos de extensibilidade da UML, mais especificamente permitindo a modelagem das atuais representações narrativas por meio de diagramas de atividade estereotipados, garantindo uma representação mais estruturada que possa ser utilizada em outros passos do processo de desenvolvimento de sistemas sem, no entanto, perder a capacidade de se comunicar com o usuário através de mecanismos que permitam gerar uma linguagem que ele compreenda.

2 Descrição de Casos de Uso

Usualmente a descrição dos casos de uso é realizada de forma textual sem nenhum compromisso com uma estrutura de representação mais formal ou estruturada [25]. Este tipo de descrição não pode ser utilizado de forma adequada por mecanismos automatizados que auxiliem o desenvolvimento de sistemas em outras etapas do processo de construção, por não possuir uma estrutura ou modelo de representação adequado.

O formato de uma descrição de casos de uso é dividido em diversas seções, que variam para cada organização, mas as seções mais comuns de serem encontrados em templates de descrições de casos de uso são [5]:

1. Atores: seção com a descrição do atores envolvidos no caso de uso;
2. Pré-condições: regras de estado que definem como o sistema deve se encontrar antes de ocorrer o caso de uso;
3. Fluxo de eventos: interações que ocorrem entre os atores e o sistema à medida que interagem para atingir um determinado objetivo;
4. Pós-condições: regras de estado que definem como o sistema deve se encontrar depois de executado o caso de uso.

A seção mais importante de uma descrição de caso de uso é o fluxo de eventos entre o ator e o sistema. O fluxo de eventos é um conjunto seqüência de eventos do tipo ação-reação que descrevem os caminhos percorridos pelo ator do caso de uso para atingir o seu objetivo.

Um exemplo simples de descrição é mostrado na Tabela 1, na qual o caso de uso "User Login" é descrito através de uma tabela que separa o fluxo entre ator e sistema [23].

Ator	**Sistema**
1. Preenche o usuário e a senha.	
	2. Valida o usuário e a senha.
	3. Realiza o login do usuário.
Alternativa 1: O usuário e senha estão inválidos. Informa o usuário sobre o erro e requisita novamente as informações de login.	

Tabela 1 – Descrição do caso de uso "User Login"

As descrições de casos de uso podem possuir uma visão externa ou interna do sistema. Em uma visão externa (ou caixa-preta) somente as atividades que são visíveis aos atores externos são descritas sem indicação de como o sistema faz para obter o resultado de uma ação. Na visão interna (ou caixa-branca) o comportamento interno do sistema é descrito. A visão externa permite a validação dos requisitos com o usuário mas pode acabar perdendo requisitos importantes que poderiam ser obtidos através de uma descrição mais detalhada. A visão interna é de grande utilidade nas próximas fases do desenvolvimento do sistema, mas difícil de ser validada com o usuário [3] devido ao detalhamento utilizado nesta visão.

Le Roy Mattingly e Harsha Rao [18] propuseram os casos de colaboração, que são casos de uso estereotipados para a descrição interna do sistema. A vantagem é a manutenção e refinamento interno dos casos de uso (casos de colaboração/visão interna) uma vez que a interface com o usuário esteja bem definida (casos de uso/visão externa).

3 Descrição através de Diagramas de Atividade

Diversas notações podem ser utilizadas para a representação de casos de uso como, por exemplo, redes de Petri, linguagens formais, pseudo-código, entre outros [25]. Em UML, diagramas de atividades, diagramas de seqüência e diagramas de estado podem ser utilizados para descrever casos de uso.

Os diagramas de atividades se apresentam como uma notação mais clara para a representação de paralelismos, elementos de lógica condicional e sincronização do que os diagramas de seqüência e diagramas de estado [3]. A descrição através de diagramas de atividade torna a descrição um artefato mais formal e estruturado, porém pode ser de difícil entendimento para os usuários e sua validação junto aos mesmos pode ficar prejudicada.

Entretanto, uma ferramenta CASE a partir de uma representação estruturada como essa poderia extrair uma descrição textual fácil de ser compreendia por usuários. A mesma ferramenta pode utilizar essa descrição estruturada para extrair outros artefatos úteis durante o processo de desenvolvimento, como por exemplo, casos de teste, casos de colaboração, métricas de complexidade, diagramas de navegação, entre outros.

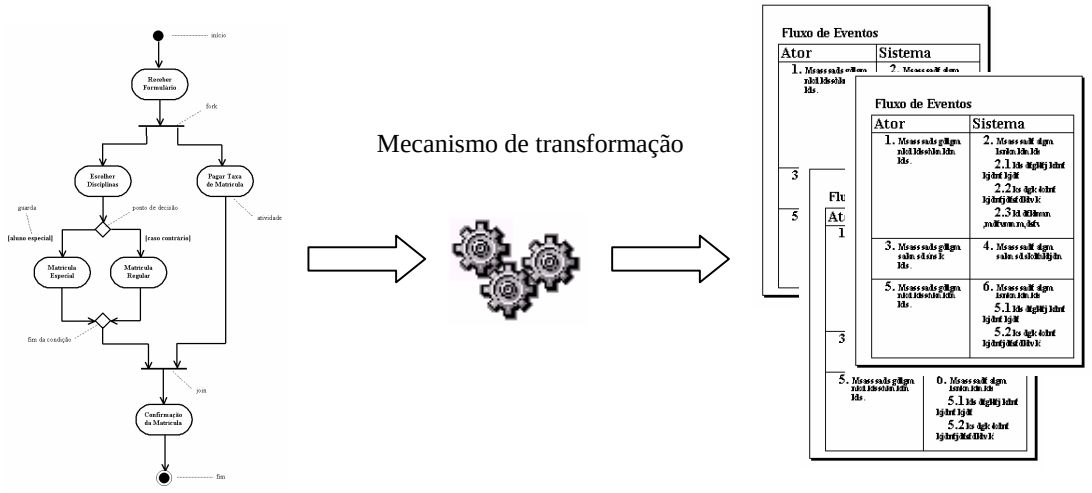


Figura 1 - Mecanismo de transformação

A utilização de diagramas de atividade para a representação de descrições de casos de uso precisa ser adaptada para suportar algumas características próprias das descrições dos casos de uso. As adaptações necessárias – como, por exemplo, a caracterização de atividades que representam passos realizados pelos atores ou pelo sistema dentro de um fluxo de eventos – podem ser realizadas através do mecanismo de extensibilidade da UML [20].

3.1 Representação do fluxo de eventos

O fluxo de eventos de uma descrição de caso de uso representa a interação entro o ator e o sistema. Essa interação se dá através de eventos numerados representados textualmente através de um fluxo contínuo ou de um fluxo separado por colunas [3][23].

Cada evento é realizado pelo ator ou pelo sistema e representa uma interação do tipo ação-reação [20]. A representação através de atividades dentro de um diagrama de atividades pode ser realizada através da utilização dos estereótipos <<ator>> e <<sistema>> para identificar o responsável pela atividade. A descrição dos casos de uso permite a introdução de seqüências alternativas, que representam desvios no fluxo principal. Essas seqüências alternativas podem ser utilizadas para realizar o tratamento de exceções ou indicar um outro caminho possível. A Figura 2 mostra como o exemplo mostrado na Tabela 1 pode ser representado através de um diagrama de atividade que utilize estereótipos.

A representação de seqüências alternativas ou fluxos condicionais é representada através de transições com guardas [4].

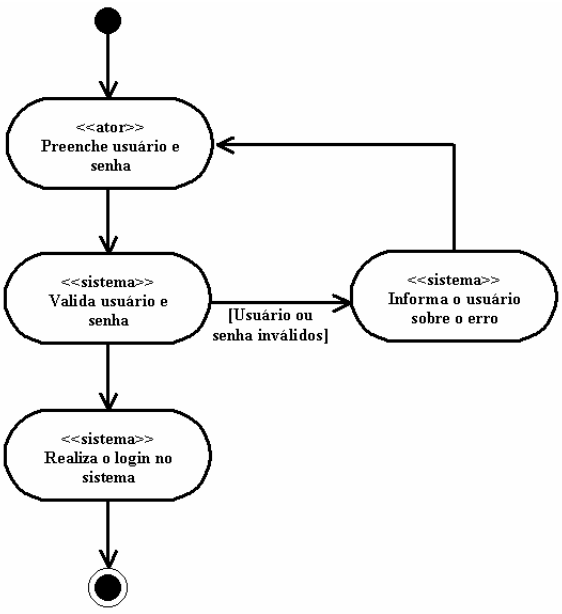


Figura 2 – Descrição através de diagramas de atividades do caso de uso "Login User"

Os diagramas de atividades possuem apenas um estado inicial e podem possuir um ou mais estados finais. Os estados finais são utilizados para indicar os possíveis términos a que a execução do caso de uso pode levar. As pré-condições e pós-condições do sistema são representadas como atributos dos elementos que representam respectivamente os estados iniciais e finais do diagrama de atividade.

Os diagramas de atividade suportam outros elementos como barra de paralelismo e de sincronização, utilizadas para representar fluxos que podem ocorrer em paralelo. As descrições textuais não possuem uma forma simples de representar comportamentos paralelos.

3.2 Representação de casos de colaboração

Os casos de colaboração são casos de uso estereotipados que descrevem o funcionamento interno do sistema [18]. A especificação de UML 1.5 possui o conceito de sub-diagramas de atividade. Essa característica permite que uma atividade possua um outro diagrama de atividade associado dentro dela. Utilizando esse conceito é possível representar casos de colaboração como sub-diagramas das atividades estereotipadas como <<sistema>>.

O caso de uso mais abstrato ou de maior nível pode definir apenas o comportamento geral do caso de uso. Durante o processo de desenvolvimento do sistema, durante a especificação mais especifica dos requisitos, o caso de uso pode ir sendo refinado através de um maior detalhamento das atividades do sistema com o uso de sub-diagramas que representem casos de colaboração.

A navegação de forma hierárquica permite que novos casos de colaboração sejam incorporados à descrição inicial do caso de uso. A validação dos requisitos com o usuário pode ser feita através da descrição mais alto-nível (primeiro diagrama na hierarquia), enquanto que os outros níveis serviriam de apoio às outras etapas envolvidas no ciclo de desenvolvimento do sistema.

O nível de detalhamento por ir aumentando à medida que novos níveis de descrição são incorporados. Um exemplo de como os casos de colaboração podem ser utilizados como diagramas de atividades estereotipados é mostrado na Figura 3.

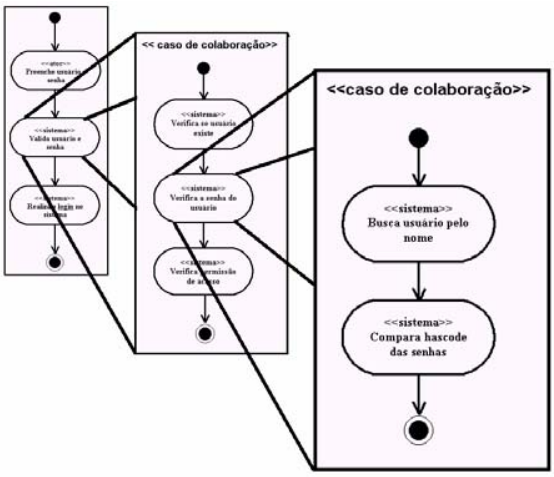


Figura 3 - Representação de casos de colaboração

Uma ferramenta CASE facilitaria o processo de descrição através de mecanismos de navegabilidade drill-down e drill-up. A descrição mais abstrata (alto-nível) poderia ser utilizada para a validação da interação do usuário com o sistema enquanto que a descrição mais específica poderia detalhar o funcionamento interno do projeto.

3.3 Representação de relacionamentos entre os casos de uso

Durante a análise são identificados os casos de uso que compõe o sistema e estes são organizados dentro de um modelo de casos de uso, onde se representam as relações dos atores com os casos de uso, e as relações entre os casos de uso. Os relacionamentos que podem ocorrer entre os casos de uso são os de inclusão, extensão e generalização/especialização [10][11][20].

O relacionamento de inclusão representa uma relação na qual o fluxo de eventos do caso de uso incluído é executado durante o fluxo de eventos de um caso de uso base. Essa chamada deve ser indicada dentro da descrição [3]. Essa indicação é representada através de uma atividade estereotipada como <<ponto de inclusão>>, e existe uma referência entre ela e o caso de uso que está sendo incluído.

Já o relacionamento de extensão apresenta dentro do fluxo de eventos indicações dos lugares onde o comportamento pode ser estendido por outros casos de uso. Essas indicações são chamadas de pontos de extensão [3] e são representadas por atividades estereotipadas como <<ponto de extensão>>. Cada ponto de extensão pode ser referenciado externamente por outro caso de uso.

O relacionamento de generalização/especialização representa a herança de comportamento entre os casos de uso. Nesse caso o fluxo de eventos é herdado e o seu comportamento pode ser alterado ou incrementado [3][11]. Este tipo de relacionamento é pouco utilizado e seu controle é mais complexo, embora possa simplificar as descrições de casos de uso que possuam pequenas alterações de comportamento. Uma ferramenta CASE poderia auxiliar na manutenção de descrições para casos de uso especializados[1].

A descrição de forma estruturada permite validar o modelo de casos de uso garantindo a consistência entre o que é modelado (relacionamentos de inclusão e extensão entre os casos de uso) e as descrições dos mesmos, uma vez que esses relacionamentos são referenciados na descrição. A relação entre a descrição estruturada e o modelo de casos de uso pode ser vista na Figura 4.

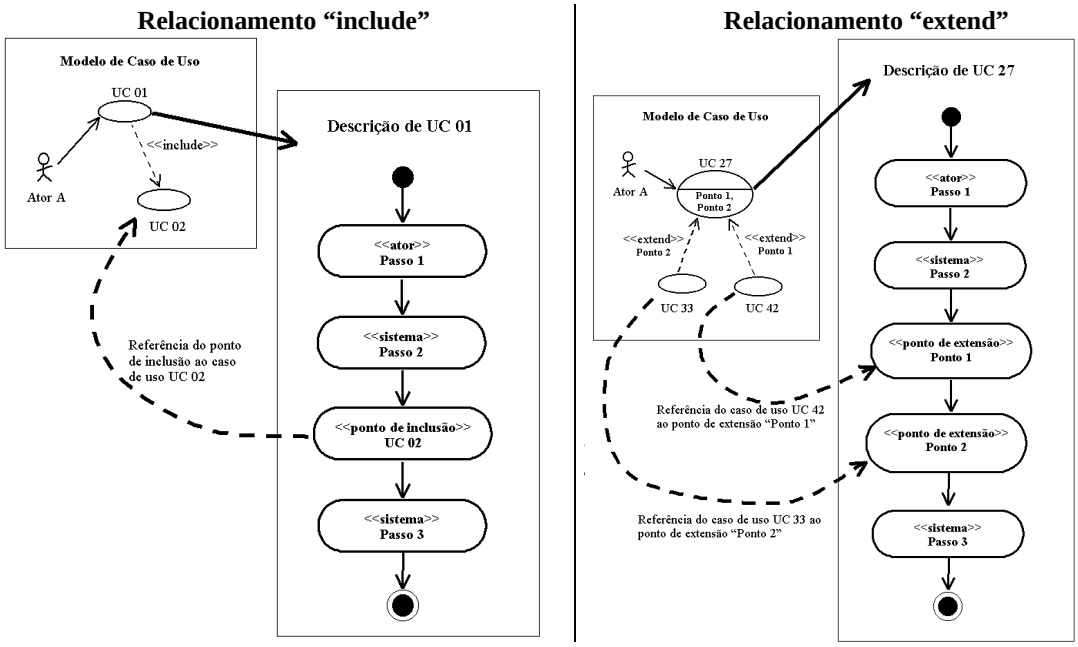


Figura 4 - Representação do relacionamento "include" e "extend"

[1] Este trabalho não contempla a descrição de casos de uso especializados, pelo fato de seu uso ser pouco utilizado [3][6][25].

4 Relação com a arquitetura e o modelo estático do sistema

A arquitetura do sistema pode ser definida como o conjunto de decisões significativas tomadas sobre a organização do sistema, a escolha dos elementos estruturais e interfaces que compõe o sistema [26]. O modelo de visão 4+1 proposto por Kruchten [16] identifica os casos de uso como a interligação entre as diversas visões concorrentes que compõe a arquitetura.

A definição da arquitetura ocorre já nas primeiras fases do ciclo de desenvolvimento. Os casos de uso iniciais ajudam a definir o que seria uma arquitetura adequada para o sistema, o suficiente para poder colocar um protótipo de arquitetura a suportar os casos de uso das primeiras iterações, dentro de um processo de desenvolvimento iterativo [15][19].

À medida que as descrições dos casos de uso e de colaboração vão se detalhando cada vez mais, as descrições entram no universo do projeto de sistemas e acabam se relacionando com a arquitetura definida para a construção do sistema.

A descrição de casos de uso através de diagramas de atividade pode utilizar o conceito de *swim lanes* [10] para identificar estilos arquiteturais (como camadas, sub-sistemas ou componentes do sistema) e identificar os eventos ou atividades associados a cada elemento da arquitetura.

Por exemplo, uma descrição de um caso de colaboração através dos diagramas de atividade poderia possuir as seguintes swim lanes: camada de apresentação, camada de negócio, camada de persistência e camada de banco de dados. Dentro de um mesmo diagrama é possível representar um fluxo de eventos e indicar em que sub-sistemas ou camadas cada evento é realizado.

Na Figura 5 pode ser visto um exemplo no qual o aprofundamento das descrições leva à necessidade de se identificar em que camada cada evento está sendo realizado. Esse tipo de associação pode ser realizado nas etapas de desenvolvimento, através da qual os próprios desenvolvedores ou projetistas podem dar continuidade à descrição criada na análise, refinando-a cada vez mais.

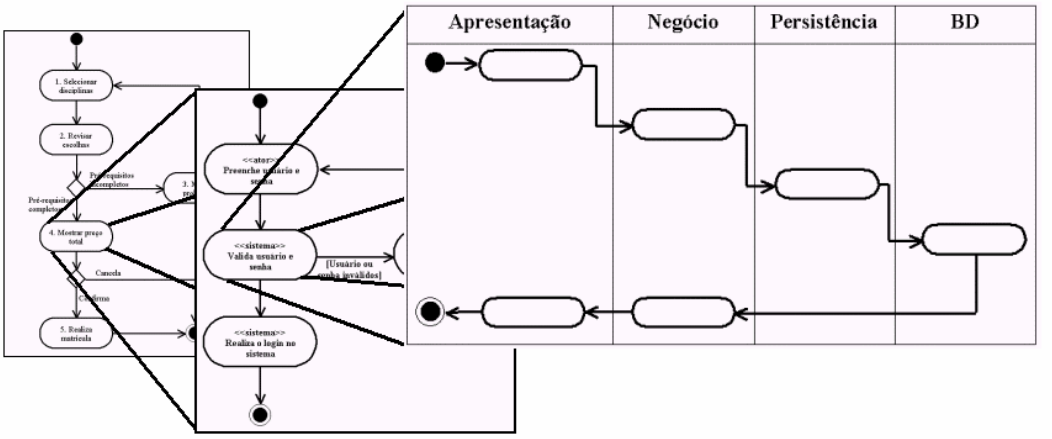


Figura 5 - Exemplo de detalhamento de uma descrição até a arquitetura

Durante o processo de análise de sistemas o modelo de classes vai sendo construído, inicialmente com classes que representam os conceitos de negócio, e posteriormente nas etapas de projeto e desenvolvimento o modelo é enriquecido com classes de projeto mais voltadas à forma como o sistema será implementado, e fortemente ligado à arquitetura definida.

Uma ferramenta CASE pode permitir a associação de elementos da descrição (eventos do fluxo de eventos do caso de uso) a classes do modelo estático ou a métodos de classes. Essa associação permite o

rastreamento de conceitos afetados por alterações de requisitos. Se, por exemplo, o usuário quiser alterar alguma funcionalidade é possível identificar os pontos afetados por tal alteração de forma automática.

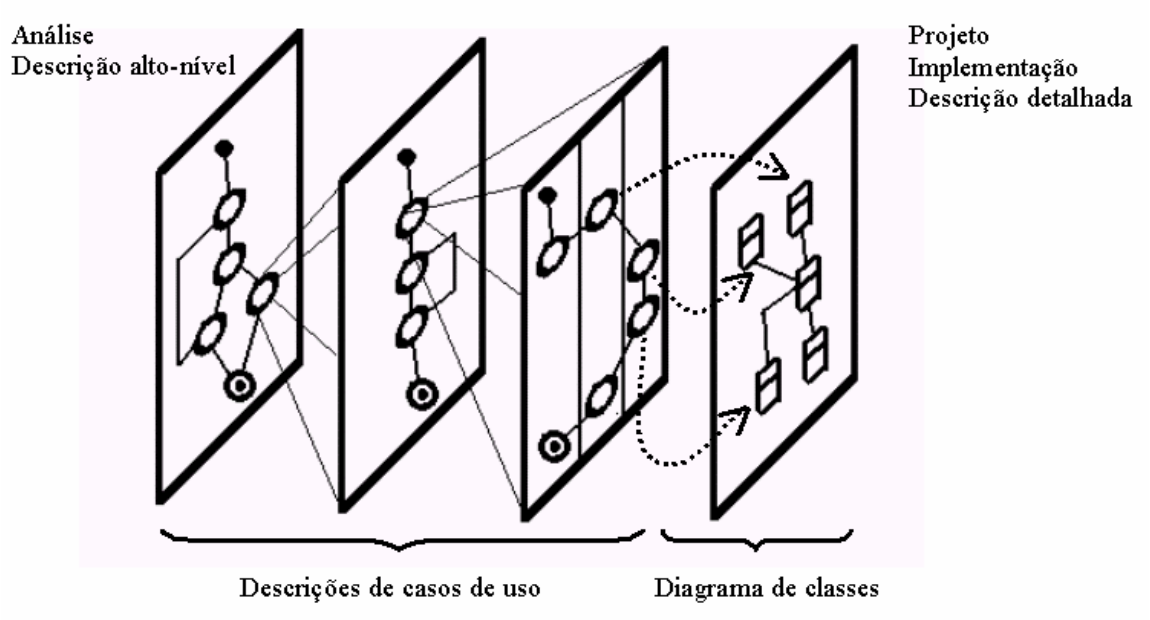


Figura 6 – Refinamento das descrições, relação com a arquitetura e modelo de classes

A Figura 6 mostra como o refinamento das descrições pode evoluir até o ponto de se ter elementos de descrição associados a classes ou métodos.

4 Modelo para a descrição dos casos de uso

A solução proposta de descrição dos casos de uso através de diagramas de atividade estereotipados é mapeada para as classes a seguir. A intenção não é a construção de um modelo que represente as estruturas definidas pela UML. A OMG já apresenta um modelo de mapeamento muito mais genérico para a sintaxe da UML do que o apresentado neste trabalho. O modelo aqui proposto é mostrado na Figura 7, definido para provar o conceito de descrição de casos de uso através de uma estrutura mais formal, baseada em diagramas de atividade. Outra modelagem seria possível, inclusive uma que reaproveitasse a modelagem definida pela OMG.

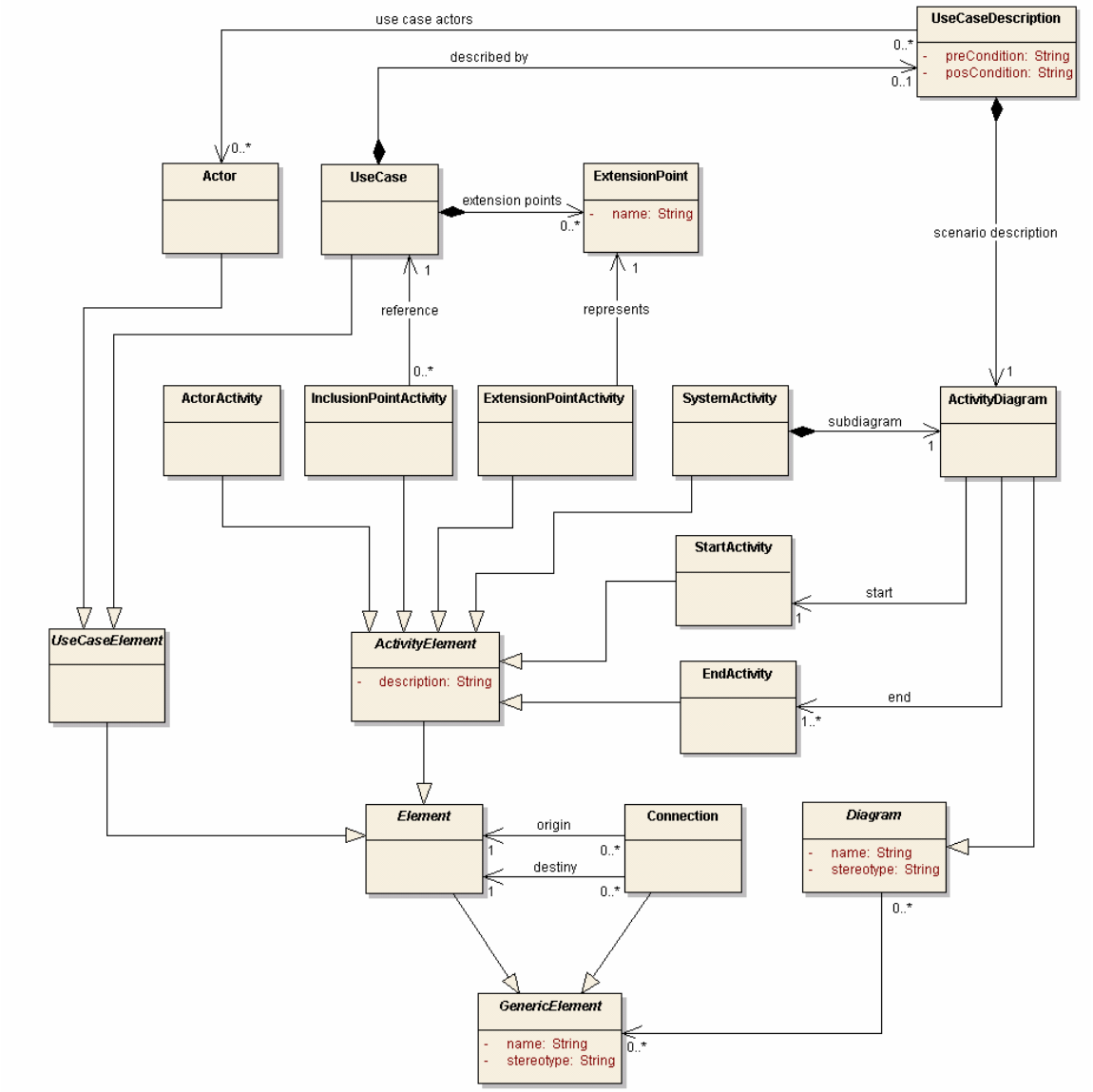


Figura 7 – Modelo geral para representação de descrições de casos de uso através de diagramas de atividade

O relacionamento da descrição com a arquitetura ocorre quando o diagrama de atividade que descreve o caso de uso possui swim lanes associadas às partições ou camadas de uma descrição de arquitetura definida. São identificadas as camadas onde cada atividade da descrição está localizada bem como as classes ou métodos aos quais a atividade está associada. A associação entre a atividade da descrição e classes ou métodos possui uma indicação sobre o tipo de referência realizado (o tipo de referência pode ser "utiliza", "executa", entre outros). A Figura 8 mostra o modelo para essas características.

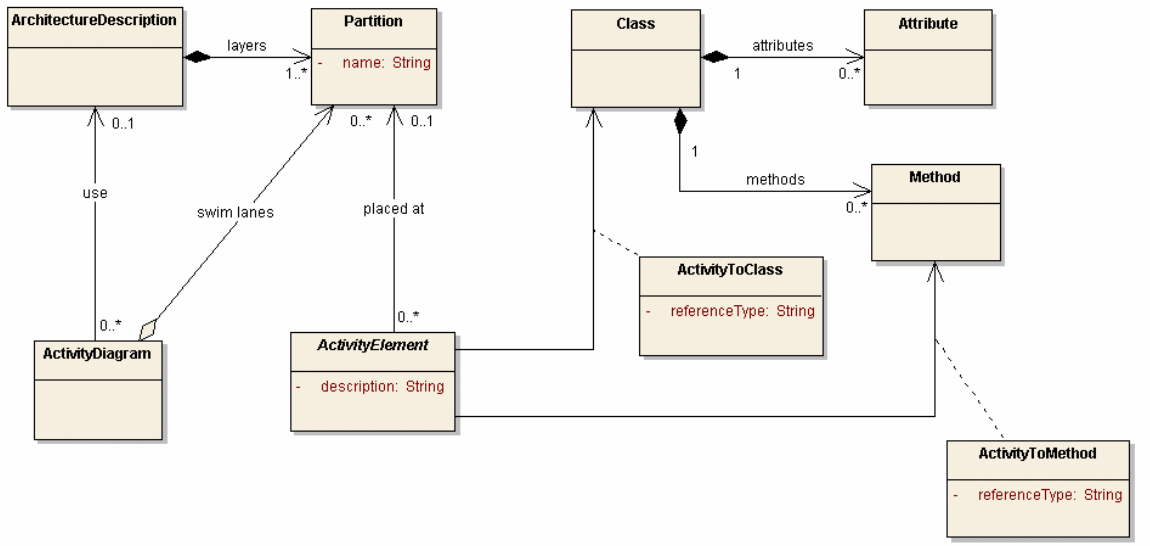


Figura 8 – Relação com a arquitetura e o modelo de classes

A Figura 9 mostra um exemplo de como uma atividade pode estar associada a uma classe e a métodos de classes. A relação com o modelo estático do sistema ocorre nos níveis mais detalhados das descrições de casos de uso.

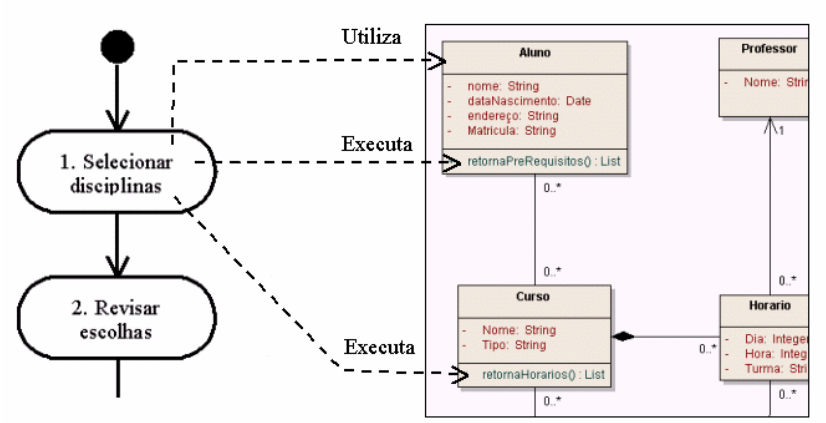


Figura 9 – Associação entre os elementos de descrição e o modelo estático

No exemplo, as referências entre as atividades e o modelo de classes do sistema (classes e métodos) possuem uma classificação cuja semântica poderia ser definida livremente pelos usuários de uma ferramenta CASE com suporte a esse tipo de descrição.

A maioria das ferramentas CASE de modelagem UML encontradas no mercado possuem funcionalidades que permitem associações simples entre casos de uso e artefatos de documentação externos (arquivos), geralmente utilizados para descrever os primeiros de forma narrativa sem nenhum comprometimento com uma estrutura de representação. Algumas ferramentas como, por exemplo, Visual Paradigm [27], possuem meios um pouco mais elaborados que permitem a descrição dos casos de uso dentro da própria ferramenta. Outros projetos como, por exemplo, o desenvolvido pela PUC-Rio chamado C&L (Cenário e Léxico) [28], auxiliam a descrição colaborativa de cenários introduzindo uma organização da informação de forma estruturada. Entretanto, essas ferramentas não permitem a associação de elementos internos das descrições dos casos de uso (passos / atividades) com outros elementos que compõe o modelo do sistema, nem possuem mecanismos para a geração de outros artefatos.

Neste trabalho é apresentada uma ferramenta CASE desenvolvida com o intuito de oferecer um mecanismo de descrição baseado no método apresentado, utilizando diagramas de atividade estereotipados para estruturar as descrições de casos de uso, permitindo o processamento para a geração de artefatos e a associação entre outros elementos do modelo do sistema.

5 Ferramenta CASE de descrição

Para a prova de conceito do método de descrição de casos de uso através de diagramas de atividade com o mecanismo de extensibilidade da UML, foi desenvolvida uma ferramenta para a construção de descrições desse tipo baseado no modelo proposto. A ferramenta se chama UC Papyrus, tem valor apenas acadêmico e possui licença GPL (GNU Public License). A ferramenta e os códigos fontes estão disponíveis em http://sourceforge.net/projects/ucpapyrus.

A ferramenta é um editor diagramático de descrições de caso de uso baseadas em diagramas de atividade. As atividades estereotipadas foram transformadas em ícones para facilitar a leitura do diagrama. A Figura 10 mostra a aparência da ferramenta.

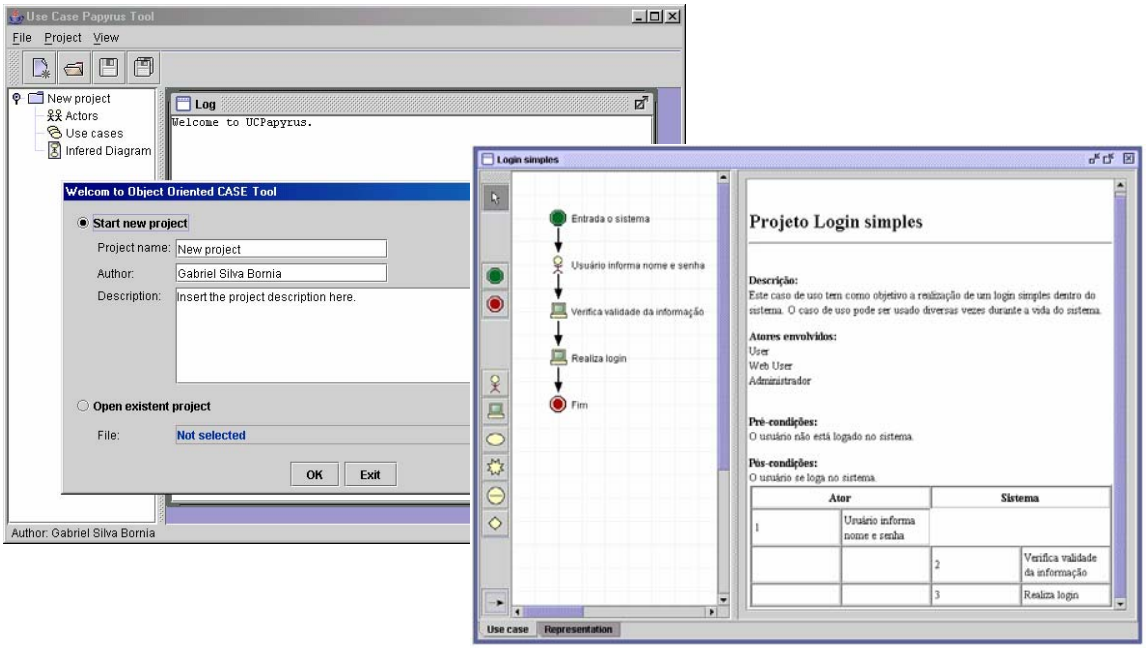


Figura 10 – Ferramenta CASE implementada

Todas as descrições de casos de uso são internamente representadas através de documentos semi-estruturados (XML). Através de regras de transformação XSL é possível processar o documento de representação e gerar diversos artefatos como, por exemplo, uma descrição amigável ao usuário semelhante às descrições textuais normalmente utilizadas no processo de análise. A ferramenta permite a adição de arquivos das regras XSL para processar as descrições.

6 Considerações finais e trabalhos futuros

A importância dos casos de uso vai muito além do escopo da representação de requisitos. Os casos de uso guiam todo o processo de desenvolvimento, a concepção da arquitetura, o projeto do sistema, a

validação dos componentes implementados, a documentação do sistema, entre outras tarefas que compõe o ciclo de vida do sistema.

O fato de se utilizar uma linguagem natural para a descrição interna dos casos de uso pode ser importante para a validação com os usuários, mas é uma informação que pouco pode ser sistematizada em processos automatizados que possam reaproveitar essas informações em outras etapas do desenvolvimento de software.

Este trabalho propõe um mecanismo que estruture as descrições dos casos de uso, de forma que a própria representação dos requisitos sirva como mecanismo de validação junto aos usuários e ao mesmo tempo sirva de entrada para mecanismos computacionais capazes de utilizar essas informações para a geração de outros artefatos úteis nas demais etapas no desenvolvimento do sistema.

O uso da extensibilidade da UML é indicado como meio de utilizar a própria linguagem UML como forma de representação estruturada para descrições de casos de uso. O objetivo é mostrar que as descrições de casos de uso podem ser modeladas através da própria UML e – se devidamente assistidas por uma ferramenta CASE – servir de base para a geração de outros artefatos que auxiliem o processo de desenvolvimento de software.

Atualmente a ferramenta CASE está sendo aprimorada para suportar a geração de informações que possam auxiliar o gerenciamento de projetos. Essas informações são métricas extraídas das próprias descrições como, por exemplo, número de transações, número de pontos de inclusão, número de pontos de extensão. Esses dados serão utilizados como entrada para cálculos de estimativa de esforço para estimar tempos de implementação [2].

A estruturação de casos de uso pode permitir a realização de trabalhos como, por exemplo:

- Geração de scripts em LOTOS para realizar a simulação de passos que possam ser executados dentro de um caso de uso, de forma que esse script possa servir como base para possíveis estudos para testes automatizados;
- A existência de uma descrição diagramática para os casos de uso pode permitir a identificação de padrões de descrições de casos de uso, que eventualmente podem vir a ser catalogados;
- As descrições podem ser enriquecidas com informações que permitam auxiliar a construção de interfaces homem-máquina;
- Implementação da rastreabilidade de alterações de requisitos, indicando os artefatos e classes que devem ser re-visitados [7].

Referências

[1] Ambler, Scott W. *Documenting a use case: what to include, and why.* Ronin International. EUA, 2000.

[2] Anda, Bente; Dreiem, Hege; Sjøberg, Drag I.K.; Jørgensen, Magne. *Estimating software development effort based on use cases: experiences from industry.* 4th International Conference on the Unified Modeling Language: UML 2001. Canada, 2001.

[3] Armour, Frank; Miller, Granville. *Advanced use case modeling: software systems.* Addison-Wesley. EUA, 2000.

[4] Booch, Grady; Rumbaugh, James; Jacobson, Ivar. *The Unified modeling language user guide.* Addison-Wesley. EUA, 1999.

[5] Cockburn, Alistair. *Structuring use cases with goal.* Humans and Technology. EUA, 2000.

[6] Cockburn, Alistair. *Use cases, ten years later – from their evolution, learn what to expect and how to better work with them.* STQE Magazine. EUA, 2002.

[7] Ecklund, Earl F.; Delcambre, Lois; Freiling, Michael. *Change cases: use cases that identify future requirements.* 11th ACM Conference on Object-Oriented Programming Systems, Languages and

Applications: OOPSLA'96. EUA, 1996.

[8] Firesmith, Donald G. *Use cases: the pros and cons. Report on Object Analysis and Design 2.* EUA, 1995.

[9] Fowler, Martin; Cockburn, Alistair; Jacobson, Ivar; Anderson, Bruce; Graham, Anderson. *"Question time! About use cases"*. 13th ACM Conference on Object-Oriented Programming Systems, Languages and Applications: OOPSLA'98. Canada, 1998.

[10] Fowler, Martin; Scott, Kendall. *UML distilled: a brief guide to the standard object modeling language.* Addison-Wesley. EUA, 1999.

[11] Génova, Gonzalo; Llorens, Juan; Quintana, Victor. *Digging into use case relationships.* 5th International Conference on the Unified Modeling Language: UML 2002. Alemanha, 2002.

[12] Hurlbut, Russel R. *The Three R's of Use Case Formalisms: Realization, Refinement, and Reification.* Expertech, Ltda. EUA, 1997.

[13] Hurlbut, Russel R. *A Survey of Approaches for Describing and Formalizing Use Cases.* Expertech, Ltda. EUA, 1997.

[14] Jacobson, Ivar; Christerson, Magnus; Patrik Jonsson; Övergaard, Gunnar. *Object-oriented software engineering: a use case driven approach.* Addison-Wesley. EUA, 1992.

[15] Jacobson, Ivar; Booch, Grady; Rumbaugh, James. *The unified software development process.* Addison-Wesley. EUA, 1998.

[16] Kruchten, Philippe. *Architectural blueprints – the "4+1" view model of software architecture.* IEEE Software. EUA, 1995.

[17] Larman, Craig. *Applying UML and patterns: an introduction to object-oriented analysis and design.* Prentice Hall. EUA, 1998.

[18] Mattingly, Le Roy; Rao, Harsha. *Writing effective use cases and introducing collaboration cases. Journal of object oriented programming 11.* EUA, 1998.

[19] Rosenberg, Doug; Scott, Kendall. *Use Case Driven Object Modeling with UML: A practical approach. Addison-Wesley. EUA, 1999.*

[20] Rumbaugh, James; Jacobson, Ivar; Booch, Grady. *The unified modeling language reference manual.* Addison-Wesley. EUA, 1999.

[21] Schneider, Geri; Winters, Jason. *Applying Use Cases: A practical guide.* Addison-Wesley. EUA, 1998.

[22] Sendall, Shane; Strohmeier, Alfred. *From use caso to system operation specifications.* 3th International Conference on the Unified Modeling Language: UML 2000. Reino Unido, 2000.

[23] Wirfs-Brock, Rebecca. *Designing scenarios: making the case for the use case framework. Smalltalk report.* EUA, 1993.

[24] Leffingwell, Dean; Widrig, Don. *Managing Software Requirements: A Unified Approach.* Addison-Wesley. EUA, 2000.

[25] Cockburn, Alistair. *Writing Effective Use Cases*. Addison-Wesley. EUA, 2000.

[26] Shaw, Mary; Garlan, David. *Software Architecture: perspectives on an emerging discipline.* Prentice Hall. EUA, 1996.

[27] Visual Paradigm for UML. http://www.visual-paradigm.com. Acesso em 10/02/2004.

[28] Cristoph, Roberto; Felicíssimo, Carolina; Leite, Julio. *C&L: Uma ferramenta de edição e visualização de cenários e léxicos*.PUC-RJ. http://sl.les.inf.puc-rio.br/cel/. Acesso em 10/02/2004.

Infraestructura de clave pública en un ccTLD empleando al DNS

Pablo Greenwood, Rolando Chaparro, Benjamín Barán
Universidad Nacional de Asunción, Centro Nacional de Computación
Asunción, Paraguay, cc1439
{pgreen, rfox, bbaran}@cnc.una.py

Resumen

La mayoría de las implementaciones de infraestructura de clave pública (PKI) aún carecen de soluciones satisfactorias en la provisión de servicios de directorios escalables para el almacenamiento y localización de certificados. En tal sentido y en relación a los requerimientos de una PKI, el sistema de nombres de dominio (DNS) presenta algunas importantes características que pueden ser utilizadas para este propósito. Además de plantear las conveniencias del DNS como servicio de directorio simple para una PKI, este artículo propone extender el vínculo DNS-PKI integrando las operaciones de delegación de dominios en un *country top level domain* con la solicitud de certificados digitales, a partir de las notables coincidencias que se pueden encontrar en ambos procesos.

Palabras claves: Redes, Seguridad de Datos, DNS, PKI, Criptografía

Abstract

Most of today's Public Key Infrastructure (PKI) implementations still face some challenges to provide scalable directory services that allow locating and retrieving certificates. However, the Domain Name System (DNS) presents a number of benefits compared to current PKI solutions, namely those based on LDAP. This paper identifies the advantages of using DNS to provide simple PKI directory services. It also outlines the use this approach to integrate a PKI certificate request and issueance with a domain name delegation process in a DNS country top level domain, which is feasible considering the significant similarities found between both procedures.

Keywords: Networks, Data Security, DNS, PKI, Cryptography

1 Introducción

El progresivo aumento de la digitalización está cambiando la manera en que se relacionan las personas y las organizaciones. La naturaleza inmaterial e impersonal de las comunicaciones electrónicas ha creado la necesidad de implementar mecanismos que permitan, cuando menos, comprobar la identidad de los interlocutores y la veracidad de los datos transmitidos, sobre todo en los sistemas abiertos como Internet.

La infraestructura de clave pública o PKI (*Public Key Infraestructure*), basada en la criptografía de claves asimétricas y en los conceptos de certificados digitales y autoridades de certificación, es una alternativa para que las aplicaciones provean servicios de seguridad como la autenticación, la confidencialidad, la integridad de los datos y el no rechazo de su origen [18].

Sin embargo, y a pesar de la madurez de la tecnología de clave pública, la mayoría de las propuestas e implementaciones de PKI presentan aún ciertos inconvenientes [5]. Entre los aspectos que aún necesitan soluciones satisfactorias, se encuentra la provisión de servicios de directorio que puedan brindar mecanismos simples de localización y recuperación de certificados.

Una PKI provee normalmente acceso a sus certificados mediante LDAP (*Lightweight Directory Access Protocol*) [21], derivado de X.500 [2]. Ambos, X.500 y LDAP, están basados en sistemas centralizados, por lo que presentan desventajas de escalabilidad en términos de recursos computacionales, requerimientos de conectividad y carga administrativa [2]. Dadas estas consideraciones, un modelo no centralizado como el DNS es preferible.

En consecuencia, este trabajo muestra que el DNS (*Domain Name System*) [10], que es un sistema distribuido, simple, tolerante a fallos y ampliamente probado en la Internet, se ajusta a los requerimientos básicos y que su uso para proporcionar facilidades de una PKI no solo es técnicamente factible a través del registro denominado CERT [3], si no que también resulta realizable en términos prácticos.

Se plantea además potenciar y extender el vínculo DNS-PKI, mediante la integración de los servicios de generación y provisión de certificados en una PKI con el registro de nombres de dominio bajo un ccTLD (*country-code Top Level Domain*) [13], debido a que se pueden encontrar notables coincidencias en los procesos para delegar un dominio y obtener un certificado. Se debe considerar que actualmente las autoridades certificadoras recurren a referencias indirectas[1], como ser los administradores de dominios, para comprobar la veracidad de los datos suministrados en las operaciones de solicitud de certificados.

En la sección 2 de este artículo se presentan las características más importantes de la infraestructura de clave pública. En la sección 3 se define la estructura del sistema de nombres de dominios y algunas de sus características que pueden ser utilizadas en una PKI. La sección 4 presenta las ventajas del protocolo DNS respecto a LDAP para almacenar certificados. En la sección 5 se identifican los factores por los cuales se puede asumir que las funciones de los ccTLDs pueden estar asociados a los procesos de certificación desde el punto de vista operacional y del DNS. La sección 6 incluye las conclusiones y algunas futuras actividades que pueden surgir a partir de este trabajo.

2 Estructura y operación de las PKIs

2.1 Modelo de organización

Las técnicas de clave pública requieren de la utilización de certificados y los correspondientes repositorios para almacenarlos. Los certificados vinculan los datos del subscriptor con su clave pública, por lo que deben tener alta disponibilidad y facilidad de recuperación.

Las recomendaciones propuestas por la IETF para una PKI (llamadas PKIX [1]), son directa derivación de los certificados X.509, los cuales se han convertido en un estándar de facto. A los efectos prácticos se identifica a PKIX como X.509 [5].

La organización de los certificados X.509 es jerárquica, utiliza una RCA (*Root Certification Authority*) en la cual se basa la confianza, que a su vez podría ser trasferida verticalmente a los usuarios mediante otras CAs (*Certification Authorities).* Específicamente la clave pública de la RCA es conocida por los usuarios. Este conocimiento es utilizado para construir un camino confiable en la jerarquía de los certificados [2].

Existen además otros modelos de certificación como PGP (*Pretty Good Privacy*), en los cuales la confianza se construye horizontalmente, entre usuarios, y no a través de las Autoridades Certificadoras. Los mismos no pueden ser implementados a gran escala, debido a la complejidad que presupone la verificación de la confianza [8].

2.2 Espacio de nombres de los certificados

Los certificados X.509 fueron desarrollados para dar seguridad al servicio de directorio X.500, pero su uso no está asociado exclusivamente al mismo [3]. La identificación de los certificados X.509 se fundamenta en el "*distinguished name*" (DN) [7].

El DN está formado jerárquicamente por un conjunto de atributos abreviados, que corresponden al país (C, *country*), la entidad (O, *organization*), a la unidad dentro de la organización (OU, *organization unit*) y la persona, máquina, etc. (CN, *common name*). Un certificado X.509 representa la asociación del DN (C+O+OU+CN) con la correspondiente clave pública del subscriptor, más otros datos relacionados [7].

La versión 3 de X.509 permite nombres "alternativos" [6], con lo cual, para identificar certificados se pueden utilizar direcciones IP, direcciones de correo, URLs, entre otros, de forma similar a lo utilizado en el DNS.

1 Referencias de otras organizaciones que prestan algún servicio al solicitante

Debido a que las CAs manejan sus propias políticas de asignación de nombres, la misma entidad podría tener el mismo DN en diferentes CAs, o distintos DNs, en distintas CAs, lo cual es ambiguo y puede representar un problema de interoperabilidad, o al menos, para la identificación unívoca de los subscriptores [5].

2.3 Servicio de directorio y protocolo de acceso

A fin de establecer una comunicación segura, las usuarios requieren buscar y obtener los certificado de otras entidades en los servicios de directorios. Por lo general las PKIs que utilizan X.509 almacenan sus certificados y listas de revocación (CRL)[2] en repositorios X.500 y como protocolo de acceso utilizan LDAP [2]. El modelo X.500 define el protocolo DAP (*Directory Access Protocol)* que requiere muchos recursos computacionales para su funcionamiento. LDAP opera sobre TCP y provee la funcionalidad de DAP pero de forma más eficiente [21]. La "L" en LDAP proviene de *Ligth* DAP.

LDAP permite almacenar y mantener completa información sobre cada una de sus entradas, en uno o más servidores. La información es almacenada jerárquicamente y los datos pueden ser actualizados mediante un mecanismo de autenticación llamado SASL (*Simple Authentication and Security Layer)* [12].

Como se puede notar, LDAP no ha sido definido para almacenar exclusivamente certificados y de hecho se lo utiliza normalmente dentro de las organizaciones para diversas aplicaciones, como por ejemplo, para repositorio de red de datos personales, para autenticación centralizada, como *yellow pages* (paginas amarillas) [22], entre otros.

2.4 Mantenimiento y operación

Una PKI tiene varios componentes que están relacionados en su funcionamiento. En una PKI participan:

- Las CAs que generan los certificados. El registro puede hacerse en una RA (*Registration Authority).*
- Los subscriptores que poseen un certificado.
- Los usuarios que recurren a las CAs para validar la clave pública de los subscriptores.

Si bien el estándar X.509v3 referencia algunas reglas o CPS (*Certification Practice Statement*) para el proceso de obtención de un certificado, en la práctica cada CA define sus propias reglas de validación [5]. Por ejemplo, es usual que una CPS acepte referencias indirectas de otras organizaciones que prestan algún servicio al solicitante del certificado [20].

En general, es posible obtener certificados de empresas comerciales internacionales. Si se requiere usar una PKI para aplicaciones legales en un contexto nacional, las operaciones y otros controles definidos en un CPS, así como otras operaciones de las PKIs, deberán estar reglamentados por las leyes gubernamentales. Difícilmente, estas reglamentaciones puedan ser consensuadas con prestadores no nacionales del servicio, lo que dificulta su utilización específicamente en dominios como el de PY.

Debido a la estructura jerárquica, las RCAs normalmente se certifican a sí mismas, o dependen de otra RCA que a su vez se autocertifíca [5], por lo cual la confianza en la jerarquía de las PKIs se basa en la autovalidación de la unidad certificadora raíz. En la Figura 1 se presenta un diagrama abreviado sobre la relación de los participantes de una PKI, asumiendo la participación de una RA, conforme se explica a continuación:

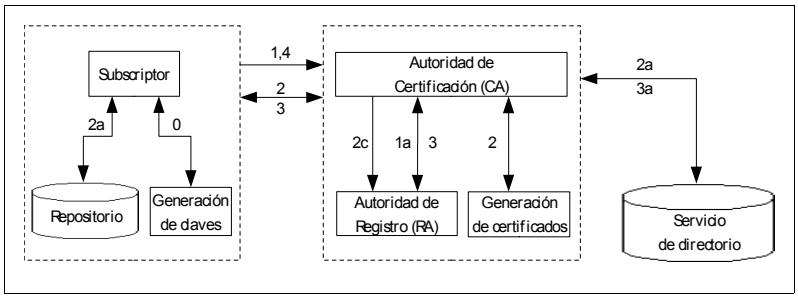


Figura 1:Organización de una PKI

2 *Certificate Revocation List*: lista para mantener los certificados que por alguna razón han sido anulados.

- *Generación de claves(0):* las entidades subscriptoras generan un par de claves, una pública y otra privada. Almacenan la privada en un repositorio seguro. La clave pública será utilizada para la generación del certificado.
- *Solicitud del certificado(1):* lo cual es posterior a la creación de las claves. Esta solicitud puede estar delegada en las Ras (1a) o ser realizadas directamente por las CAs. En este punto, la CA o RA, debe verificar la validez de los datos enviados en la solicitud por parte de las entidades.
- *Generación del certificado(2):* posterior a la verificación, el certificado es generado por la CA. La CA puede solicitar al subscriptor la comprobación de su clave pública. El certificado es almacenado en el servicio de directorio (2a) y enviado al subscriptor través de la RA si existe (2c).
- *Revocación de certificados(3):* en el caso que la clave privada del subscriptor se encuentre comprometida, el certificado deberá ser revocado (3a). También la CA podría revocar el certificado si es que comprueba que los datos proveídos no cumplen reglas de la CA.
- *Actualización de certificados (4):* generada por el vencimiento del certificado o por el cambio de alguno de los datos asociados al mismo, puede ser originada por la CA o por los subscriptores.

Si bien algunos de estos procesos utilizan a modo de referencia los estándares de clave pública criptográfica PKCS desarrollado por RSA Inc.[3], como por ejemplo, PKCS#10 [15] o PKC#12 [16], las CAs utilizan éstos u otros procesos, según sus propias normas de certificación (CPS).

3 Sistema de resolución de nombres (DNS)

3.1 Estructura organizativa

Las personas pocas veces hacen referencia a los recursos de la red mediante direcciones IP, sino que lo hacen mediante cadenas de caracteres como el URL *http://www.cnc.una.py*. Sin embargo, la red en sí misma requiere de direcciones binarias para direccionar los recursos. Por lo tanto, se necesita algún mecanismo para convertir las cadenas de caracteres en direcciones IP.

Para resolver este problema se desarrolló el DNS [19] que es un protocolo con un esquema de nombres jerárquico, basado en dominios y en una base de datos distribuida que soporta esta organización [10]. Si bien el DNS se usa principalmente para relacionar las direcciones destino con números IP, también puede utilizarse con otros fines, como por ejemplo, para asociar certificados a éstas direcciones a través del RR (*Resource Record*) CERT [3].

La estructura jerárquica del DNS esta dividida en zonas. Cada zona contiene una parte de la estructura y a los servidores de nombres de esa zona llamados *Authoritives Names Server*. Para resolver un nombre a su correspondiente dirección IP, el cliente consulta inicialmente a su servidor de nombre local. Si el servidor conoce la respuesta, podrá retornar el resultado. Si no la conoce, enviará la consulta a los *Root Servers*[4], los cuales están ubicados en la parte superior del árbol. Estos a su vez informarán sobre los servidores de segundo nivel responsables de ese dominio. Este proceso se repetirá con los siguientes niveles del árbol hasta llegar al servidor donde se encuentran los datos sobre el dominio solicitado [19].

Con el objetivo de evitar la sobre carga de los servidores del DNS, el protocolo implementa la posibilidad de hacer *cache* de las consultas realizadas. Así, cada vez que el *resolver*[5] obtiene una referencia sobre un dominio o algún dato sobre éste, lo almacena localmente y en forma temporal. El tiempo de éste almacenamiento es controlado por el TTL (*Time To Live*) asignado a cada RR utilizado en el DNS.

En términos prácticos, se puede mencionar que el funcionamiento del DNS está altamente probado y que el mismo es una precondición para la comunicación en Internet y entre las partes de una PKI.

3.2 Alcance de los nombres de dominio

La estructura de nombres del DNS se divide en cientos de dominios de nivel superior, cada uno de los cuales incluyen todas las direcciones para ese dominio.

3 *RSA Laboratories, division of RSA Data Security Inc.*
4 Existen en la actualidad 13 *root servers* distribuidos geográficamente.
5 Servidor que atiende las solicitudes de los clientes y realiza las consultas a la base de datos del DNS.

Cada dominio se puede dividir en sucesivos subdominios. Los dominios de nivel superior pueden ser genéricos (edu, net, com, mil, org, etc.) a los cuales se los denomina gTLDs (*Generic Top Level Domains*) y de países, definidos en el estándar ISO 3166[6] a los cuales se los llama ccTLDs [13]. Cada recurso se nombra por la trayectoria ascendente en la estructura del árbol hasta la raíz, que es representada por un "." (punto). Por su organización, se puede asegurar que no existirán dos nombres completos[7] iguales bajo dominios diferentes [10].

3.3 Servicio de directorio

El DNS tiene su propia base de datos, la cual es distribuida. Cada dominio puede contener un grupo de r*esources records* [19]. Existen distintos tipos de RR, pero el más común es el que asocia una dirección IP a un nombre. Cuando un *resolver* consulta sobre un dominio, lo que recibirá será los RRs asociados al mismo, los cuales se encuentran en la base de datos del servidor de ese domino. Cada RR consta de 5 atributos: el nombre completo, el ttl, el tipo, la clase y el valor. Dependiendo del tamaño de la respuesta, la misma podrá ser enviada mediante los protocolos UDP o TCP [11].

El nombre completo representa al recurso sobre el cual se necesita información, y es la clave primaria para la búsqueda. El *ttl* es una indicación de la estabilidad del registro en el *cache* de los demás servidores DNS. El tipo referencia de que registro se trata, algunos de ellos son: NS (servidor DNS), A (dirección de IP a un *hosts*), SOA (inicio de autoridad), MX (servidor de correo del domino). La *clase*, en el caso de Internet, siempre estará representado por "IN". El *valor* que puede ser un número IP, un nombre o una cadena ASCII dependiendo del tipo de RR [19].

Se puede notar, que la base de datos del DNS no solo mantiene números IP, por lo que podría ser utilizada para almacenar también certificados. De hecho y como ya se mencionó, en las actualizaciones del DNS se ha definido un tipo de RR llamado CERT [3], para guardar certificados.

Las especificaciones de actualizaciones dinámicas (*Dinamic Updates*) [23] del DNS implementan dos mecanismos de seguridad para agregar, actualizar o eliminar registros, llamados:

- *Secure DNS Transaction and Request Authentication*
- *Secret Key Transaction Authentication*

3.4 Modelo jerárquico de delegación y administración

Los TLDs son delegados a través de los *Root Servers*, según el ente autorizado para la asignación de nombres y direcciones de Internet, de acuerdo a los principios contenidos en [13]. En general la mayoría de las organizaciones en la región que administran los dominios ccTLDs son entidades reconocidas que históricamente se encuentran relacionadas a Internet y a su desarrollo, como ser universidades (CL, CO, MX, PA, PY, UY), redes académicas (BO), dependencias del gobierno (BR, AR, CU) o en organizaciones sin fines de lucro (PE, SV, HN)[8].

Se puede suponer, que si la administración de los ccTLDs fuera delegada en otras organizaciones, éstas serán de similar categoría, asumiendo la labor que realizan y el interés de los usuarios finales acerca del buen funcionamiento de la Internet. Siempre deberá existir alguna entidad responsable de los ccTLDs.

Al igual que en las PKIs, en el funcionamiento del DNS están involucradas distintas entidades. Inicialmente se identifican a las entidades administradoras, que mantienen y operan los servidores de nombres de los ccTLDs o gTLDs. En segundo lugar, se encuentran las organizaciones que delegan sus dominios y por último los usuarios, que utilizan al DNS para buscar y obtener información sobre los recursos a fin de iniciar una comunicación. A continuación se presenta un esquema abreviado del proceso de delegación (ver Figura 2):

- *Solicitud de delegación del dominio (1):* iniciada a pedido de la entidad interesada, la cual es enviada al administrador del ccTLD.

6 Código de identificación de países compuesto de dos letras (PY, PE, BR, AR, US, BO, SV, etc.).

7 También conocido como *fully qualified domain name* (FQDN).

8 Según datos extraídos del LacTLD (*Latin American & Caribbean Country Code Top Level Domain Organization*) *http://www.lactld.org/members.html*

- *Verificación de la solicitud(2):* el administrador del dominio verifica los datos incluidos en la solicitud y envía una petición de confirmación de los mismos a la organización interesada (2a).
- *Aprobación de la delegación(3):* una vez confirmado los datos, la delegación deberá ser aprobada según las pautas del administrador. El resultado será notificado a la organización (3a).
- *Actualización de la base de datos(4):* representa la creación y activación de la delegación del nuevo dominio en la base de datos del TLD.

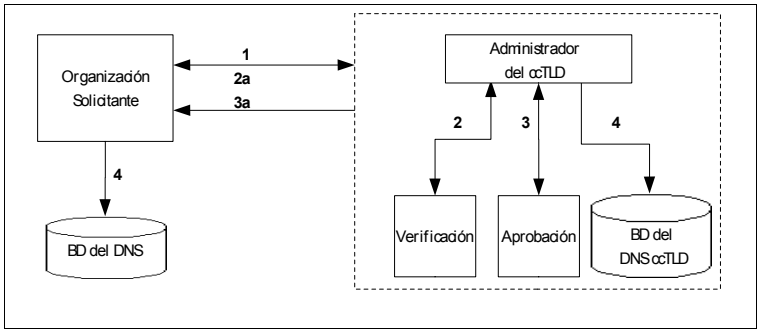


Figura 2:Proceso abreviado de delegación de dominios

En relación a la aprobación, el administrador del dominio realiza la verificación de los datos del solicitante, de forma similar a lo realizado por las CAs. En general, el nivel de verificación de la identificación es definido en cada ccTLD de acuerdo a sus propias pautas de delegación.

Se debe resaltar que en el DNS no se requiere mantener una lista de dominios eliminados o revocados, debido a que los RR asociados al dominio son retirados de la base de datos del DNS, por lo cual, las consultas realizadas sobre ese dominio no podrán ser respondidas.

4 Comparación de LDAP y DNS.

4.1 Servicios de directorios

A los efectos de realizar comparaciones de rendimiento se presenta a LDAP y al DNS como alternativas de almacenamiento, teniendo en cuenta que las implementaciones más comunes de PKI están basadas en LDAP y que a su vez el DNS permite almacenar certificados en su base de datos.

El servicio de directorio para una PKI debe soportar identificación de certificados a través de direcciones Internet, como los nombres de *hosts* o direcciones de correo. Las búsquedas que utilizan nombres comunes para la identificación de certificados carecen de validez en un ámbito general como la red Internet. Así mismo, los datos utilizados para localizar una dirección de red son los nombres de *hosts* o direcciones de correo, los cuales son únicos y están almacenados en el DNS.

Se pueden encontrar referencias sobre las deficiencias de LDAP para almacenar certificados en [2].

4.2 Localización simple

El DNS está definido para resolver un nombre de *host* a una dirección IP [11]. Cuando se contacta con un *host*, el DNS encuentra su número IP asociado, el cual será utilizado para establecer la comunicación. Se debe notar que antes de establecer la comunicación, se sabe con quién se quiere contactar. De la misma manera, pero a través del RR CERT se puede obtener el certificado asociado a ese recurso.

La localización del certificado está dada por clave (nombre de *host*, dirección de correo, etc.) la cual es única y conocida. Además, las aplicaciones no necesitan implementar protocolos distintos para la resolución de nombres y localización de certificados más que el DNS. El DNS es implementado por casi todas las aplicaciones que usan Internet.

En el caso de LDAP, las aplicaciones requerirán rutinas diferentes para la resolución de nombres y para el acceso a los certificados. Además, las CAs y LDAP no tienen definido de forma natural una única estructura raíz como el DNS, por lo que no se sabe a que servidor se debe recurrir para encontrar un certificado en particular.

4.3 Paquetes según protocolo de acceso

El DNS provee un mecanismo simple de localización de direcciones y certificados. Estas características se encuentran directamente relacionadas a la implementación del protocolo y al fin para el cual fue creado (asociar direcciones Internet con nombres de *hosts*). El DNS utiliza los protocolos UDP y TCP. Si el tamaño del paquete excede al soportado por UDP (600 bytes), el protocolo utiliza automáticamente TCP. Un paquete DNS se divide en: *header, question* y tres *answers* [10].

LDAP esta definido para dar acceso a los servicios de directorios X.500 [2], que poseen características mucho más amplias que las del DNS. Los paquetes LDAP utilizan el protocolo TCP y se dividen en: *message id*, b*ind request, bind response, unbind resquest, search request, search result entry, search result done, search result referent, modify request, modify response, controls*, entre otros [21].

Como se puede notar, la estructura de LDAP es mucho más compleja, lo cual es natural debido a los tipos de datos que soporta y al tipo de acceso que requieren sus usuarios. Por ejemplo, en el DNS las consultas no son interactivas y son realizadas por las aplicaciones. En el caso de LDAP, el acceso es interactivo para los usuarios, por lo que soporta mantener sesiones durante las conexiones [21]. En la Figura 3 se muestra el round trip[9] del DNS para realizar una consulta mediante TCP, considerando que el tamaño de los paquetes UDP por lo general no son suficientes para enviar certificados.

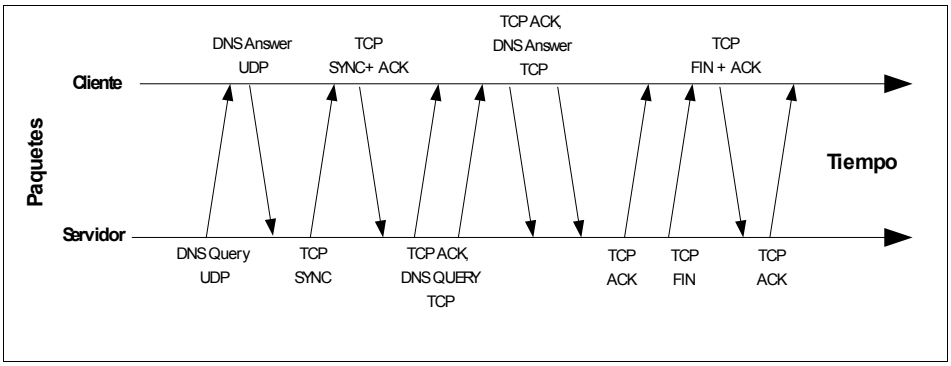


Figura 3:Consulta DNS

Como se puede apreciar, el DNS automáticamente cambia de TCP a UDP. Inicialmente envía una consulta UDP, si la respuesta no es completa, establece una conexión TCP [11].

En el caso de LDAP, luego de establecer la conexión TCP, el cliente envía un "*bind request*" para establecer la sesión, la cual se confirma en con un "*bind result*". La consulta en sí misma se efectúa mediante un "*search request*" y la respuesta se obtiene en los paquetes "*search entry*". El estado de la consulta es enviado al cliente a través de un "*search response*". Para finalizar la conexión el cliente envía un "*unbind request*" y cierra la conexión. En la Figura 4 se muestran los paquetes en una conexión LDAP.

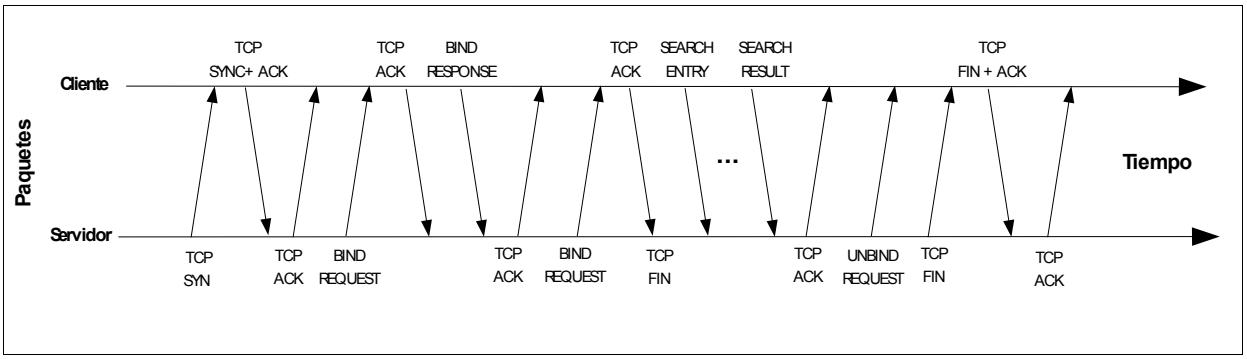


Figura 4: Consulta LDAP

Como se ve, para hacer una consulta LDAP se requieren más paquetes que en una consulta DNS. En la Tabla 1 que se muestra a continuación se resume el número de paquetes utilizados en cada caso.

9 Flujo de paquetes en el tiempo.

Tabla 1: Resumen de los paquetes DNS y LDAP

Protocolo	Cantidad de paquetes
DNS (incluyendo el fall-back)	12
LDAP	15 o más

4.4 Tamaño de los paquetes

La cantidad de paquetes requerido por el DNS para la localización y obtención de un certificado es menor. Si ambos servicios son utilizados bajo la misma infraestructura de red, el DNS también tendrá mejor rendimiento. En la Tabla 2 se muestran los resultados realizados en una red Ethernet, utilizando un certificado RSA de 1024 bits [9].

Tabla 2: Tamaño de los paquetes en DNS y LDAP

Protocolo	Tamaño en bytes
DNS (incluyendo el fall-back)	802
LDAP	852

5 Propuesta de ccTLDs como soporte de una PKI

5.1 Dependencia y uso del DNS

Como se mencionó en la sección 3.1, una PKI usualmente ve, necesita y usa al DNS como un servicio de infraestructura para establecer la comunicación entre sus entidades (CAs, RAs, y usuarios finales). Estrictamente, el DNS es fundamental para el funcionamiento de Internet pero no la PKI.

Los certificados utilizados por las organizaciones para comunicaciones externas no tienen sentido si la misma no posee un dominio (identificación) en Internet.

Así mismo, se deben considerar las funciones de los administradores de los ccTLDs [17], quienes delegan dominios a las organizaciones, las cuales podrían necesitar un certificado, y los procedimientos relacionados a dichas delegaciones, por lo que la delegación de un dominio puede estar asociada a la generación de certificados para dicho dominio.

5.2 Zona de dominio[10] de los ccTLDs

Debido al ámbito en el cuál opera el ccTLD (un país), se asume que será posible y más seguro autenticar la identidad de las organizaciones solicitantes de los dominios delegados y certificados, disminuyendo la posibilidad de errores. Se debe resaltar que una de las referencias utilizadas por las CAs para validar la identificación de las organizaciones, son los datos proveídos por los administradores de los ccTLDs, como en nuestro caso, el NIC-PY.

Así mismo, cabe enfatizar que en dominios como el de PY no existen PKIs nacionales operativas, por lo que se recurre a empresas no nacionales para obtener el servicio. Integrar ambas infraestructuras (ccTLD-PKI) representaría un importante vehículo de promoción de las PKIs y una ventaja para los usuarios.

5.3 Autoridades certificadoras

La confianza en las CAs comerciales se basa en la práctica en la autocertificación de ellas mismas. Debido a su naturaleza comercial, éstas pueden dejar de operar por diversos motivos (económicos, políticas de mercado, etc.) con lo cual los certificados y la seguridad de las conexiones estarían comprometidos.

Como se menciona en el apartado 3.4, la administración de los ccTLD es responsabilidad de organizaciones estrechamente relacionadas al funcionamiento de Internet. Si por alguna razón éstas dejaran de operar, la responsabilidad será delegada a otra organización igualmente comprometida con el desarrollo de Internet.

10 Ámbito de operación y alcance del ccTLD.

5.4 Procedimiento de delegación y certificación

Para que el ccTLD, en este caso el del dominio PY, emita certificados y brinde funcionalidades de una PKI, el presente trabajo propone integrar los procedimientos de certificación a los de delegación de dominios. De hecho, como se describe en los apartados 2.4 y 3.4, los procedimientos de certificación y delegación son similares, pero no iguales. Los procedimientos deben ser ajustados de manera a soportar la transferencia segura de las claves y datos de la solicitud, lo cual puede efectuarse mediante los estándares PKCS u otros recomendados.

Los administradores del ccTLD deben incorporar las CPS a fin de ajustar sus políticas de verificación de identidad de acuerdo a las recomendaciones existentes sobre PKIs [1].

La delegación del dominio puede o no incluir la generación de un certificado, dependiendo del requerimiento de la organización interesada. Se estima que solamente el 1.73% de *hosts* utilizan certificados[11], lo cual no representa una carga adicional muy relevante para el administrador del ccTLD, como tampoco lo es para la base de datos del DNS. En la Figura 5 se muestra un diagrama del procedimiento propuesto:

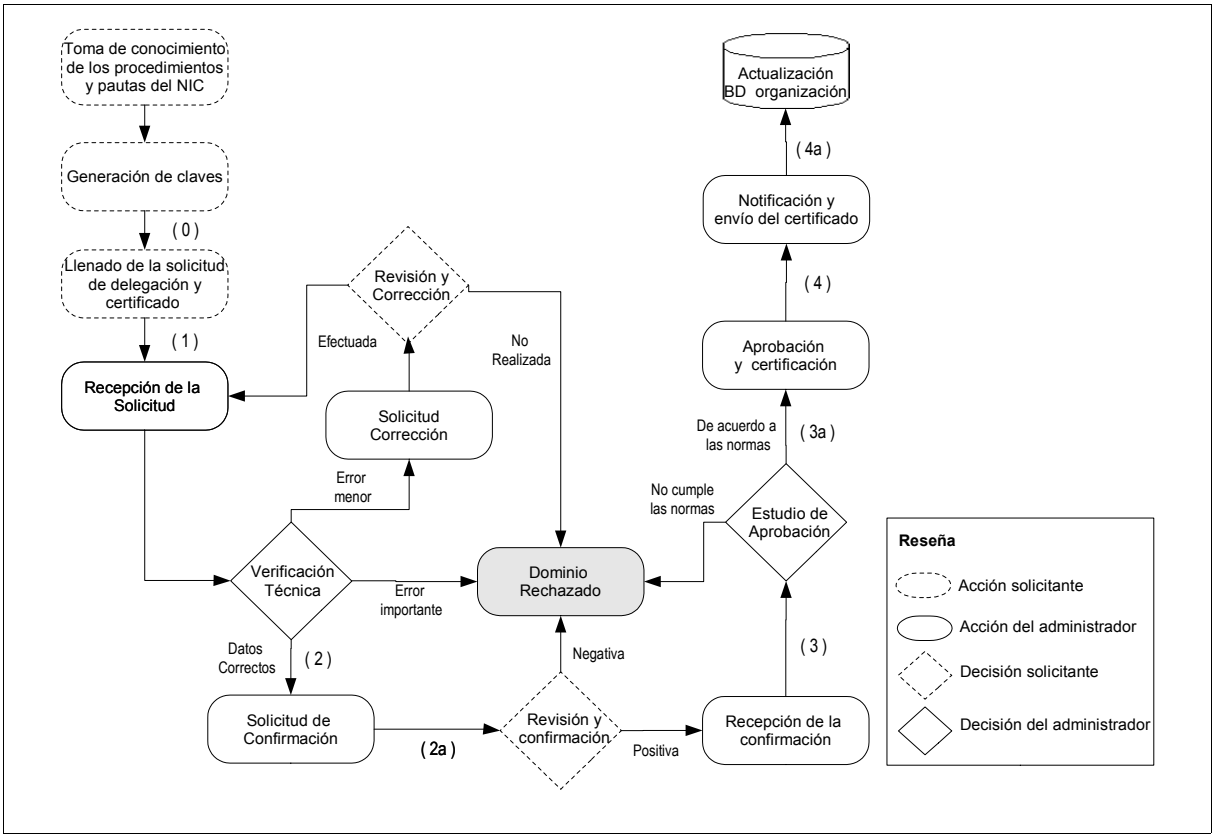


Figura 5: Proceso de delegación y certificación

A continuación se presenta un esquema abreviado acerca de las operaciones de delegación y certificación en un ccTLD:

- *La generación de claves(0):* las entidades subscriptoras generan un par de claves, una pública y otra privada. Almacenan la clave privada en un repositorio seguro. La clave pública será utilizada para la generación del certificado.
- *Solicitud de delegación del dominio y generación del certificado(1):* iniciada a pedido de la organización interesada, la cual es enviada al administrador, adjuntando la solicitud de generación del certificado a través de un medio seguro.

11 Según datos obtenidos de *http://www.securityspace.com*

- *Verificación de la solicitud(2):* el administrador verifica los datos incluídos en la solicitud del dominio y solicita confirmación de los mismos a la organización interesada, como también la confirmación de la clave privada (2a).
- *Estudio de aprobación de la delegación(3):* una vez confirmado los datos, la delegación del dominio deberá ser aprobada según las pautas del administrador (3a). Además, se debe realizar la verificación de la identidad del solicitante acordes a las leyes nacionales y recomendaciones internacionales.
- *Generación del certificado(4):* aprobada la delegación, el resultado es notificado a la organización, el certificado es generado por el administrador y enviado al subscriptor, quien deberá agregar el RR CERT en las tablas de su servidor de dominio.

La eliminación del dominio por cualquier circunstancia, implicará automáticamente la revocación del certificado de la organización, atendiendo a que éste no podrá ser encontrado por los *resolvers* en la estructura jerárquica del DNS. Un certificado no tiene sentido si no existe manera de acceder a él.

5.5 Modelo de certificación

En el *modelo básico* de delegación, a partir de la posesión del dominio, del certificado y de su correspondiente clave privada, la organización podrá solicitar actualizaciones a los datos de su dominio, como ser, el cambio de servidores.

De la misma manera, las organizaciones podrán solicitar certificados para sus *hosts* y cuentas de correos, los cuales una vez generados, deberán ser enviados a la organización solicitante y agregados en sus respectivos servidores de nombres.

Se puede además implementar un *modelo extendido*, en el cual las organizaciones (que poseen un dominio) generan sus propios certificados firmando con su clave privada, por lo que podrán administrar libremente las modificaciones, asignaciones y eliminaciones de los certificados que corresponden a su propia infraestructura.

Gracias a que en el *modelo extendido* las organizaciones generan sus propios certificados, las mismas podrán responder más rápidamente a las modificaciones de nombres de *hosts,* asignaciones IP y direcciones de correo, entre otros. Una desventaja de este modelo, es el costo adicional que implica para las consultas del DNS construir el camino jerárquico de los certificados, dado que se debe verificar el certificado de las organizaciones (que no se encuentran en el *cache*) y luego los certificados por ella firmados.

Como se puede apreciar, la integración DNS-PKI permitirá inclusive definir modelos de certificación más flexibles que posibilitarán a su vez menor tiempo en las actualizaciones y rapidez en las respuestas ante incidentes debido a la cercanía del administrador.

5.6 Revocación y naturaleza de las consultas DNS

Debido a que las consultas a través del DNS se hacen en línea mediante el *resolver* [11], los certificados serán obtenidos desde los servidores al momento que el cliente los necesite y éste podrá almacenarlos localmente para evitar acceso continuo al servidor DNS, controlando el tiempo de vida del certificado. Actualmente las CAs emiten sus listas de revocación a intervalos de tiempo definidos, según lo consideran pertinente.

Para validar la existencia del certificado se necesita simplemente realizar la consulta al *resolver* sobre el RR CERT del DNS nuevamente, por lo tanto se puede prescindir de las listas de revocación (también planteado en [14]) disminuyendo los inconvenientes relacionados a las mismas en cuanto a la operación y administración de las PKIs.

5.7 Interoperabilidad de PKIs

Es casi imposible pensar que una sola CA y su infraestructura puedan soportar el mantenimiento y administración de certificados en un contexto general, no solo por los recursos necesarios, sino también por la diversidad de requerimientos de acuerdo a cada país o región.

En el caso del modelo del DNS cada dominio es administrado internamente, por lo que se tiene una distribución jerárquica bien definida en relación a la infraestructura.

En un modelo más abierto, la confianza podría basarse entre ccTLDs, en cuanto que la responsabilidad de certificación recae sobre quién mejor conoce su zona de dominio.

5.8 Servicio de directorio

Si bien LDAP es completo como servicio de almacenamiento, las PKIs requieren acceso simple a los certificados [5]. El sistema distribuido requerido por las PKIs, presupone la cooperación entre servidores para soportar la escalabilidad. Esto es soportado y utilizado ampliamente en el DNS.

En cuanto a los procesos de adición, modificación, eliminación de los datos, se debe recordar que ambos servicios soportan protocolos seguros de actualización como se plantea en los apartados 2.3 y 3.3. En relación al rendimiento, en el capitulo 4 demuestra que el DNS es eficiente para almacenar certificados.

A estos factores se debe agregar que la gran mayoría de las aplicaciones utilizadas en Internet soportan el DNS, por lo que no requieren implementar otros protocolos para acceder a los certificados.

6 Conclusiones y actividades futuras

A partir de las deficiencias actuales de las PKIs en relación a la infraestructura y al modelo de certificación mayormente centralizados, existen básicamente dos factores muy importantes por los cuales se puede sostener la relación de los ccTLDs con una PKI:

- La dependencia operacional del DNS para el funcionamiento de toda la red
- La eficiencia del servicio de directorio del DNS y la conveniencia de su espacio de nombres

Como se demostró, es completamente factible la integración de los procesos de delegación y obtención de certificados en los ccTLDs, lo que representa una importante ventaja desde el punto de vista de los usuarios. Se debe considerar que los procesos asociados a ambas infraestructuras en realidad son uno solo, ¿ de qué sirve un certificado si no se tiene un dominio ?.

Debido al ámbito nacional en el que opera un ccTLD, los administradores pueden adaptar sus políticas de delegación a las leyes nacionales del país, lo que facilita la relación con los subscriptores. Así mismo se asume que es posible y más seguro autenticar la identidad de las organizaciones solicitantes disminuyendo la posibilidad de errores, teniendo en cuenta que la responsabilidad recae sobre quién mejor conoce su zona de dominio. De hecho, en la actualidad las CAs recurren a los administradores de los ccTLD para validar el espacio de nombres y la autenticidad del solicitante. Asociar el proceso de delegación de dominios y certificación permitirá además promover el uso de las PKIs.

Por otra parte, los usuarios en general y una PKI en particular, necesitan y utilizan al DNS como un servicio de infraestructura para establecer una comunicación. El DNS es fundamental para el funcionamiento de Internet y como servicio de directorio provee un mecanismo simple de localización de direcciones y certificados (el cual está dado por una clave única y conocida) requisito fundamental en la infraestructura de clave pública.

Por la naturaleza en línea de las consultas al DNS, los certificados son obtenidos cuando se realiza la conexión y pueden ser fácilmente actualizados, lo cual implica un mayor grado de seguridad en relación a las listas de revocación, atendiendo que las mismas actualmente son generadas en plazos definidos según las políticas de cada CA.

Además, el protocolo DNS está ampliamente probado, es eficiente, jerárquicamente distribuido, tolerante a fallos e implementado en casi todas las aplicaciones que usan Internet, por lo que no se necesita implementar otros protocolos adicionales para acceder a los certificados.

Como trabajo futuro, los autores esperan especificar y desarrollar los entornos operativos para la administración de los certificados por parte del ccTLD, así como también definir los prototipos de aplicaciones que efectúen consultas tipo CERT en el DNS para posibilitar la integración de este modelo.

Referencias

1. Chokhani S., Ford W., Sabett R., Merrill C., y Wu S. *X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework*. IETF, RFC 3647. (Noviembre 2003).
2. Chadwick, D. *Deficiencies in LDAP when usde to support PKI*. Communications of the ACM, Vol. 46, No. 3. (Marzo, 2002).
3. Eastlake, D. *Storing Certificates in the Domain Name System*. IETF, RFC 2538. (Marzo, 1999).
4. Eastlake, D. *Domain Name System Security Extensions*. IETF, RFC 2535. (Marzo, 1999).
5. Gerck, E. *Overview of Certification Systems*, The Bell, Vol. 1, No. 3. (Julio 2000).
6. ITU-T Recommendation. *X.509v3 Information Technology Open Systems Interconnection – The directory: public-key and attribute certificate frameworks*. ITU-T. (Mayo, 2001).
7. ITU-T Recommendation. *X.501 Information Technology Open Systems Interconnection – The Directory: Models*. ITU-T. (1993).
8. Josang, A., Pedersen, I. y Povey, D. *PKI Seeks a Trusting Relationship*. Springer-Verlag. (Julio, 2000).
9. Josefsson, S. *Network Application Securtity Using The Domain Name System*. Master's thesis, Stockholm University. (2001).
10. Mockapetris, P. *Domain Names - Concepts and facilities*. IETF, RFC 1034. (Noviembre, 1987).
11. Mockapetris, P. *Domain Names - Implementation and Specification*. IETF, RFC 1035. (Noviembre, 1987).
12. Myers, J. *Simple Authentication and Security Layer (SASL)*. IETF, RFC 2222. (Octubre, 1997).
13. Postel, J. *Domain Name System Structure and Delegation*. IETF, RFC 1591. (Marzo, 1994).
14. Rives R. *We Can Eliminate Certificate Revocation Lists*. Springer-Verlag. (Febrero, 1998).
15. Nystrom M, y Kaliski B. *PKCS #10: Certification Request Syntax Standard Version 1.7*. IETF, RFC 2986. (Noviembre, 2000).
16. RSA Laboratories. *PKCS #12: Personal Information Exchange Syntax Version 1.0*. RSA Laboratories. (Junio, 1999).
17. Stahl, M. *Domain Name Administrators Guide*. IETF, RFC 1032. (Noviembre, 1987).
18. Stallings, W. *Cryptography and Network Security, 2nd edition*. Prentice Hall. (1999).
19. Tanenbaum, A. *Computer Networks, 3rd edition*. Prentice Hall. (1997).
20. VeriSign. *Normas para el proceso de certificación para los servicios de certificación bajo la VeriSign Trust Network Version 2.0*. CertiSur. (Julio, 2003).
21. Wahl, M., Howes, T. y Kille S. *Lightweight Directory Access Protocol Versión 3*. IETF, RFC 2251. (Diciembre, 1997).
22. Wang, X., Schulzrinne, H., Kandlur, D. y Verma, D. *Measurement and analysis of LDAP performance*. ACM Press. (2000).
23. Wellington, B. *Secure Domain Name System (DNS) Dynamic Update*. IETF, RFC 3007. (Noviembre, 2002).

Experimental Studies Using SOARA: An Approach to Reduce Alarm Rates on Streams of Intrusion

Jorge Levera
jlevera@cs.uic.edu

Robert Grossman
grossman@uic.edu

University of Illinois at Chicago
Department of Computer Science
Chicago, IL, 60612, USA

and

Benjamin Barán
Universidad Nacional de Asunción
Centro Nacional de Computación
San Lorenzo, Paraguay
bbaran@cnc.una.py

Abstract

The overwhelming number of alarms generated by rule-based network intrusion detection systems makes the task of network security operators ineffective. Preliminary results on an approach called SOARA shows that false positive alarms can be reduced by detecting changes on streams of alarms using sketch-based time-decaying moving median. SOARA keeps a memory efficient sketch summary of the normal stream of alarms using relevant features. Sketches are updated according to established policies and a time-decaying moving median procedure is used on historical data to detect abnormal alarm rates on the stream. SOARA shows promising results on labeled and unlabeled test sets by focusing on exceptions on the normal stream of alarms, diverting the attention away from false positives.

Keywords: Data stream, intrusion detection, sketch summaries, time-decaying moving median.

1 Introduction

Given the proliferation of valuable assets on the Internet, it has become clear that network security operators are looking for robust intrusion detection systems (IDS) that are efficient, effective and easy to manage. Roughly speaking, IDS can be classified in host-based IDS (HBIDS) and network IDS (NIDS). HBIDS monitors users pattern of behavior on a particular host, and try to detect abnormal sequence of commands, or vulnerability exploits. NIDS monitors network packets in order to detect unusual or potentially hazardous traffic.

A NIDS can categorized as rule-based (RBIDS), statistical-based (SBIDS) or protocol based (PBIDS) intrusion detection system. RBIDS inspect network packets passing through the network and compare them with a set of rules. A match triggers an alarm. SBIDS build a statistical model of the network behavior, and trigger alarms for every deviation from the model. PBIDS trigger alarms for every departure from standard network protocols.

The most popular approach is RBIDS [9] because it is very effective against known attacks and efficient when the set of rules is kept to a reasonable size. Its main drawbacks are the impossibility of detecting unknown attacks and the overwhelming amount of alarms that could be generated [9, 10].

A great number of alarms generated by RBIDS are false positives [5, 7, 10] (i.e., attack did not actually take place). False positive alarms could be reduced by adding customized filters to the IDS or by eliminating the rules that causes the noise. Sometimes, it is difficult to apply any of those changes because either the IDS belongs to another organization (e.g., outsourcing, cooperative distributed IDS) or it is not safe to delete a specific rule. Then, the security operator is faced with the problem of detecting true positive alarms among a pile of false positives.

Several approaches have been proposed to reduce the number of alarms. Solutions were proposed from the realm of data mining [13, 14, 17], machine learning [20] and visualization [10, 26]. This paper presents a Stream Of Alarms Reduction Architecture (SOARA) to reduce intrusion alarms using change detection algorithms over streams of intrusion alarms.

In the context of SOARA, stream of alarms generated by IDS arrive to a SOARA trimmer which reduces them to simple and expressive time series. Those time series or signals are used by SOARA analyzer to build sketch-based summaries that model the behavior of the stream of alarms [11,16]. Changes on the stream of alarms are detected using a time-decaying moving median procedure [6] on signal values kept in the sketch. The signals that deviate significantly from the normal pattern of behavior are used to generate a predictable and manageable number of generalized alarms. Those trend alarms help network security operators focus on the most interesting data. Experimental results on labeled and unlabeled datasets show that SOARA can detect changes on the trend of alarms by focusing on exceptional data.

The rest of the paper is organized as follows: Section 2 surveys related work on alarm reduction. Section 3 presents the SOARA approach to alarm reduction. Section 4 shows experimental results. Section 5 discusses future work and concluding remarks.

2 Related work

Data mining techniques has been applied to intrusion detection data for several years [17, 21, 16, 28]. Lee and Stolfo [17] were the first ones to use clustering and association rule on system and network features in order to learn their normal behavior. Manganaris et al. [21] used alarm features to characterize the normal stream of alarms using association rules. Ye and Li [28] used clustering and classification on alarm features. Most of the previous approaches require a carefully selected training set in order to build a model of normal behavior and classify unseen data. Portnoy et al. [23] pioneered on unlabeled intrusion datasets and proposed a clustering technique to detect intrusions. Likewise, SOARA works on unlabeled data and requires no training.

Julisch and Dacier [14] presented a different approach to alarm reduction based on conceptual clustering techniques. In a later work, Julish [15] used clustering to find the root of most false positives. Shortcomings of their approaches include periodic tuning to adjust the model to changing network conditions, and the numerous parameters that are not trivial to determine [5]. Though SOARA requires a couple of tuning paramenters, they can be found and tuned easily requiring infrequent adjustments.

Lately, several authors have proposed techniques to correlate alarms [7, 22, 25]. Valdes and Skinner [25] provided a mathematical framework for alert correlation, extending ideas from multisensor data fusion. Cuppens and Miege [7] clustered, merged and correlated alarms in a cooperative IDS environment. Ning et al. [22] built attack scenarios. They correlated alarms by partial match of prerequisites and consequences of attacks. Correlation condenses alarms to a few groups and facilitates the distinction between false and true positives. However, prerequisites and consequences conditions are not trivial to find.

In the context of streaming data, several change detection algorithms for streams has been studied. Gilbert et al. [11] proposed sketch summaries base on wavelets to approximate aggregate data from streams. Barford et al. [5] reported results of signal analysis of four classes of network traffic anomalies using wavelet filters over streams. Krishnamurthy et al. [16] proposed a sketch-based summary of streams and applied several forecasting models. The authors in [5, 16] presented a general model that could be adapted to alarm reduction. Their experimental work is base on streams of network packets and flow information rather than streams of alarms. In [16] they mentioned false positive reduction as on going work based on the top n most abnormal streams. SOARA extends these ideas to the context of rule-based IDS where the input is a stream of alarms instead of network flow data. SOARA incorporates memory efficient implementation of their algorithm based on cache memory policies. Besides, sketch summaries used in SOARA store historical data at different time intervals in order to represent individual stream of alarms more accurately.

There are several other related work in the context of stream mining. Cohen and Strauss [6] formalized the problem of time-decaying aggregates and statistics over streams of data. Dong et al.[8] discussed mining of changes from data streams and proposed the expiration of old data based on data's distribution instead of arrival time. Yamanishi and Takeuchi [27] presented a framework for detecting outliers and change points on non-stationary time series data. Babcock et al. [3] presented techniques for maintaining variance and k-median clustering over data stream windows. SOARA brings together several of the previous mining techniques to the context of intrusion alarm reduction.

3 Reducing alarms with SOARA

This Section presents SOARA, a sketch-based time-decaying moving median approach to reduce alarm rates based on change detection over streams of intrusion alarms. First, the model used to summarize the stream is presented. Then, the time-decaying moving median approach is used to detect changes on the normal stream of alarms.

Throughout this paper, IDS alarms are assumed to be a set of n attributes $A=(a_0, a_1, ..., a_{n-1}) \in \Omega$ that describe a particular alarm and the network packet that triggered it. Common attributes are: time of the event, identifier of the rule that triggered the alarm, source IP address, target IP address, and target port number to name a few. For instance, Table 1 depicts a set of alarms generated by Snort IDS [24] where the first row defines an alarm $A_0 = (a^0_0=1; a^0_1=$"2004-03-12 10:24:35"$; a^0_2=$"SNMP trap tcp"$; a^0_3=58; a^0_4=40; a^0_5= 5912; a^0_6=162) \in \Omega$.; the second to fifth rows define A_1 to A_4 in a similar way.

Table 1. A set of intrusion alarms A_i from a Snort IDS and their corresponding attributes a^i_j

a_0 = rule Id	a_1 = Time Stamp	a_2 = Description	a_3 = Time-to-live	a_4 = IP length	a_5 = IP Id	a_6 = Target Port Number
1	2004-03-12 10:24:35	SNMP trap tcp	58	40	5912	162
2	2004-03-12 10:24:37	SCAN Proxy (8080) attempt	48	40	57215	8080
3	2004-03-12 10:24:39	SNMP request tcp	38	40	9835	161
4	2004-03-12 10:24:40	SCAN SOCKS Proxy attempt	38	40	59362	1080
2	2004-03-12 10:24:41	SCAN Proxy (8080) attempt	128	40	57310	8080

3.1 Modeling normal stream of intrusion alarms

There are several approaches to model streams of data [16]. In SOARA alarms arrive sequentially as a stream of events $A_0, A_1, A_{2, \ldots}$ to the SOARA trimmer. The goal of the trimmer is to reduce the number of attributes in A by combining them or discarding. The SOARA trimmer build streams of features $K = <k_0, k_1, .., k_L>$ where $K=F(A)$ is a function of subset of attributes in A. The stream $K_0, K_1, K_2, \ldots$ is used by SOARA analyzer in order to build sketch summaries of the stream at predefined time intervals. Figure 1 shows the architecture behind SOARA and its components. For simplicity, in this paper the knowledge database associated with SOARA is not used.

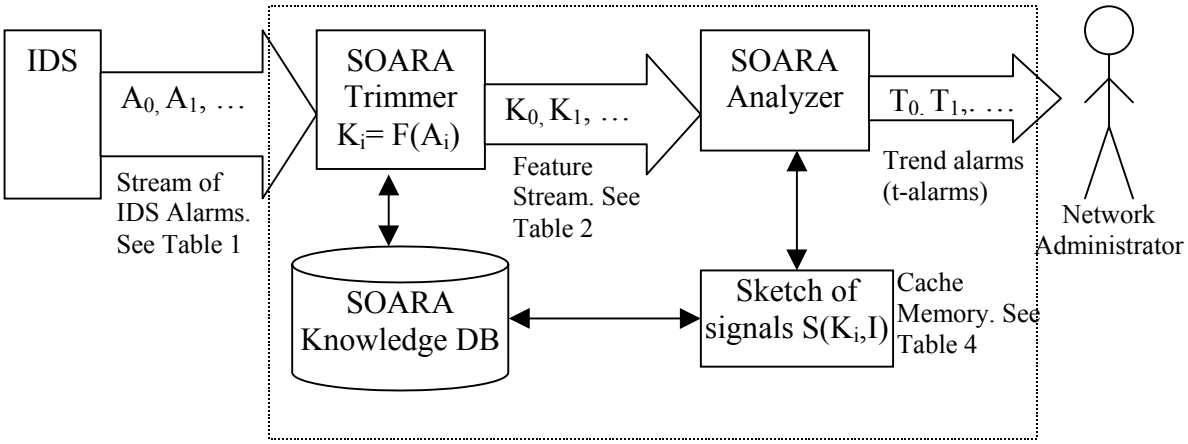


Figure 1. SOARA architecture

For the purpose of this paper, the attributes used in K are: *(k_0* =identifier of rule that triggered alarm, k_1=destination port number, k_2=time-to-live attribute of network packet that triggered the alarm*)*. The attributes in K were chosen based on previous experiments with sets of alarms [18]. Table 2 shows a stream of feature vectors $K_0, K_1, K_2, \ldots$ occurring at descrete time intervals t_i. The first row represents the feature vector $K_0 = < k^0_0 = 1; k^0_1 = 58; k^0_2 = 162 >$ that occurred at time $t=485$ minutes, the second row represents a feature vector $K_1 = < k^1_0 = 2; k^1_1 = 48; k^1_2 = 8080>$ that occurred at the descreate time $t=487$, and so on.

Table 2 Feature vectors K_i built by SOARA trimmer

Discrete Time	k_0=Rule Id	k_1=Time-to-live	k_2=Target Port Number
485	1	58	162
487	2	48	8080
489	3	38	161
490	4	38	1080
491	2	128	8080

During consecutive time intervals of size t called $I_0, I_1, I_2 \ldots$ the SOARA analyzer receives streams of feature vectors $K_0, K_1, \ldots$ from the SOARA trimmer. The analyzer summarizes the alarm feature vectors received during each I_i, and builds signals or time series of the form $S(K, I_i) = u_i$ where u_i is an update value during the interval I_i. In this paper, the update value u_i correspond to the rate of the feature vector K_j during the interval I_i. Table 3 shows five instances of signals $S(K_i, I_0)$ where $I_0=[485, 500]$ is a 15-minute interval. The first row represents the signal $S(<485, 1, 58, 162>, I_0) = 0.27$, the second one signal $S(<487, 2, 48, 8080>, I_0) = 0.33$, and so on.

Consequently, for each value K_i there is a time varying signal $S(K_i, I)$. The arrival of a new alarm A_i during the interval I_j causes the signal $S(K_i, I_j)$ to be updated with $S'(K_i, I_j) = S(K_i, I_j) + u_i$. The goal of SOARA is to identify all those signals $S(K_i, I_j)$ with significant changes in their normal pattern of behavior. The fact that k_1 correspond to a rule identifier for this particular implementation of SOARA, relates signals $S(K, I)$ and intrusion alarms A (i.e., $k_0=a_0$). For this reason, in this paper we use the term signal rate and alarm rate interchangeably.

Table 3 Signals S(K_i, I) within a 15-minute interval I_i

Discrete Time	k_0=Rule Id	k_1=Time-to-live	k_2=Target Port Number	u_i=Rate per minute
485	1	58	162	0.27
487	2	48	8080	0.33
489	3	38	161	0.27
490	4	38	1080	0.27
491	2	128	8080	0.07

In SOARA, sketch summaries are implemented by storing a set of m distinct signals $S(K_i, I)$, $0 \leq i \leq m$ that are identical in term of space and structure allocated in memory. Associate with each signal $S(K_i, I)$, SOARA stores the time interval I_j of last update to the signal, and historical data u_i.

The historical data field stores the last w update values ($u^i_0, u^i_1, \ldots u^i_{w-1}$) of signal $S(K_i, I_j)$ during different intervals I_j. Note that the historical data ($u^i_0, u^i_1, \ldots u^i_{w-1}$) correspond to observed values of signal $S(K_i, I_j)$ at different, possibly not consecutive, time intervals I_j. The only temporal restriction regarding two consecutive values u^i_j and u^i_{j+1} is that u^i_j was observed during an interval I_h prior to the interval I_n where u^i_{j+1} was observed. Table 4 shows sketch summaries of signals $S(K_i, I)$ as they are represented in SOARA. Each row correspond to a signal $S(K_i, I)$ for a specific value of K_i. The first column stores the last update interval time I_i for signal $S(K_i, I)$ represented in discrete time according to the value of interval I_i in minutes. For instance, the first row correspond to $K_0=(k^0_0=1, k^0_1=58, k^0_2=162)$ implying that signal $S(<1,58,162>, I)$ has been

updated for the last time during interval *I=[1140, 1155]* minutes, and has five values of u^0_i (i.e., historical data) stored in chronological order (0.13, 0.07, 0.13, 0.07, 0.13, 0.13).

Table 4. A snapshot of sketch summaries used in SOARA. Rows represent signals *S(K,I)* and are identified by the combination of rule id, target port number and time-to-live of the network packet that triggered the alarm. For each signal, a windows of size *w* stores the signal's rate

Discrete Update Time I	*Rule id*	*Time-to-live*	*Target Port Nbr*	Rate 0	Rate 1	Rate 2	Rate 3	Rate 4	Rate 5
1155	*1*	*58*	*162*	0.13	0.07	0.13	0.07	0.13	0.13
1245	*2*	*48*	*8080*	0.47	0.40	0.20	0.20	0.40	0.07
1275	*3*	*38*	*161*	0.07					
1350	*4*	*38*	*1080*	0.20	0.40	0.20	0.20	0.20	0.80
1260	*2*	*128*	*8080*	0.7					

As mentioned before, SOARA keeps track of a maximum of *m* signals $S(K_i, I)$ at the same time. In order to use memory efficiently, signals $S(K_i, I)$ are treated like cache entries in a cache memory. At the beginning, the first $S(K_i, I)$ distinct signals are tracked and stored in memory. Their respective entries u_i are updated as new alarms arrive and time intervals I_i increase. When the *m+1* distinct signal arrives, a replacement algorithm is used to allocate a new entry according to a predefined policy.

In this paper, SOARA uses a replacement policy based on time of last update and priority assigned to cache entries. For instance, when the cache is full (i.e., there are m distinct signals been tracked), the signal that has not been updated for the longest time is removed. If there is more than one candidate entry for replacement, then the entry with the lowest priority is replaced.

Signal priorities are assigned according to the rule id related to the signal. Most IDS have a priority associated with the alarms generated by it. The priority identifies the level of importance associated with a particular type of alarm. In Snort IDS, each rule has a priority *p*, $1 \le p \le 5$ associated with it where *p=1* rank the most dangerous alarm types. Consequently, every alarm generated by Snort due to a match with rule id *r* that has priority *p* will have a priority of *p*. In SOARA, every signal $S(K_i=<k_0, k_1, k_2>, I)$ where $k_0=r$ has a priority *p*. For example, if rule id four is an alarm of priority one according to the IDS, then any signal $S(K_i, I)$ where $k_0 = 4$ will have a priority of one to stay in the cache.

3.2 Change detection algorithm

In this Section, a simple approach to change detection is presented. In SOARA, a change detection algorithm computes the deviation of each sketch-summarized signal $S(K_i, I)$ with respect to a value predicted by a model. For simplicity, SOARA use moving median as the predictive model. Every *t* minutes, the value predicted by the moving median is compared against the new rate value for that particular combination of rule ID, TTL and target port number. The top *d* signals with the greatest deviation (i.e. error value) are considered trend alarms (t-alarm) because they deviate significantly from the rate values predicted.

Formally, the *w* observed values $(u^i_0, u^i_1, .., u^i_{w-1})$ of signal $S(K_i, I)$ are used as input to a predictive model $F(u^i_0, u^i_1, .., u^i_{w-1})$. The value $F(u^i_0, u^i_1, .., u^i_{w-1})$ is compared against the next

new value u^i for $S(K_i, I)$. Trend alarms called t-alarms are generated for the top d values $e_i = u^i - F(u^i_0, u^i_1,.., u^i_{w-1})$. Intuitively, t-alarms represent the top d signals $S(K_i, I)$ that differ the most from the predicted value. Note that in SOARA, positive values of e_i denote an increase on the rate of alarms related to signal $S(K_i, I)$. SOARA focuses on rate increases only because they tend to indicate the beginning of an attack or the abnormal increase of a normal noise.

```
i = 0;   S[ ]=0;
F[ ]=0;  // initialize forecast value for signals S

Forever do
        {
        While ( size_of_time_Interval(Ii) <= t )do
                {
                A =Receive_alarm();
                // aggregate to vector K[]
                Kj[ k0, k1,.. , kL ] = f(A);
                // update signal S
                if (Kj[ ] already exist) then
                        S[Kj, Ii] = S[Kj, Ii] + ui;    // update signal S
                else {
                        Allocate_space(S[ ]);  //remove oldest S[ K] if necessary
                                                // to keep track of at most m distinct signals
                        Insert_New_Signal( S[Kj, Ii] );   // keep track of new signal
                     }
                // increase time
                Update(Ii , clock() );
                }
        // compute error in signals S[K, Ii]
        compute_error(S[ ], F[ ]);

        // trigger alarm for most deviated signal S[K, Ii]
        trigger_trend_alarm(S[ ]);

        // update forecast value for signals S[ ]
        predicted_value(F[K]);

        //move to next time interval Ii
        i = i + 1;
        }
```

Figure 2. Pseudo code for SOARA

It should be noted that if error is normalized, the relative importance of each entry is eliminated. Usually, the more often a type of alarm is triggered, the more important its deviation from the normal pattern is.

The error e_i associated with a signal $S(K_i, I)$ decreases with time. Say the current time interval I_j has just finished. If the signal $S(K_i, I)$ has last been updated during interval I_h, then the error e_i associated with signal $S(K_i, I)$ is reduced to $e_i = e_i * (I_h / I_j)$. The error is recalculated at the end of each time interval, before t-alarms are generated for the d most deviated signals. Figure 2 shows the pseudo code for SOARA.

4 Experimental results

SOARA was tested with four sets of alarms generated by Snort IDS version 1.9.1 with default set of rules. One of the IDS was located on a Public Sector Metropolitan Area Network of Asuncion, Paraguay [4]. Another set of alarms was generated by a Snort IDS located on the

Abilene network [1]. An additional set belongs to an IDS running at the University of Illinois at Chicago (UIC). The last set of alarms was generated using DARPA99 Intrusion Detection Set [19]. Table 5 gives an overview of the datasets used.

Table 5. Datasets used on experimental results. Some characteristics are shown to indicate their diverse nature and evaluate the results on each set

Dataset	Network Type	Number of Alarms	Distinct Rule Ids	Time Span (days)	Distinct Source IP Addresses	Distinct Target IP Addresses
UIC Network	LAN	592,121	52	180	20,161	2,429
Pub. Sect. MAN	MAN	430,794	67	7	514	272
Abilene Network	WAN	10,357,673	49,571	90	208,527	253,790
DARPA99	Synthetic	23,050	84	10	109	187

In order to compare SOARA's behavior over different datasets, we used the same tuning parameters for all datasets. In general, different networks require different tuning paramenters that depend on the amount of alarms generated by the IDS and received by the SOARA analyzer. Besides, the size *m* of the cache needed to hold signals S(K,I) depends on the diversity of the alarms generated by the IDS influence. The more diverse the origin, number and type of alarm the more likely it is that distinct signals will need to be tracked. The size of the network monitored by the IDS also affects the size of the cache.

The following parameters were used during the experiments. Time intervals I_i were generated every 15 minutes. SOARA kept track of a maximum of $m=30$ distinct signals *S(K,I)*. For each signal *S(K,I),* SOARA kept a windows of size $w=7$ update values. Only $d=1$ t-alarm was generated for every interval I_i. The error value e_i decayed at a rate of *0.5* for every time interval I_i. In consequence, the moving median procedure ran every $t=15$ minutes over the historical data kept for every signal.

Table 6 shows the reduction obtained over the datasets. The reduction was computed as the number of RBIDS alarms to be examined on the exceptional signal pointed by t-alarms over the total number of RBIDS generated during the same time interval I_i.

Table 6. Average reduction on four datasets

Dataset	Avg. Reduction
Public Sector MAN	0.89
Abilene Network	0.78
UIC Network	0.59
DARPA 99	0.77

In general, UIC network dataset did not generated many raw alarms at $t=15$ minute intervals. In order to obtain greater reduction on these datasets, a larger *t* value is needed.

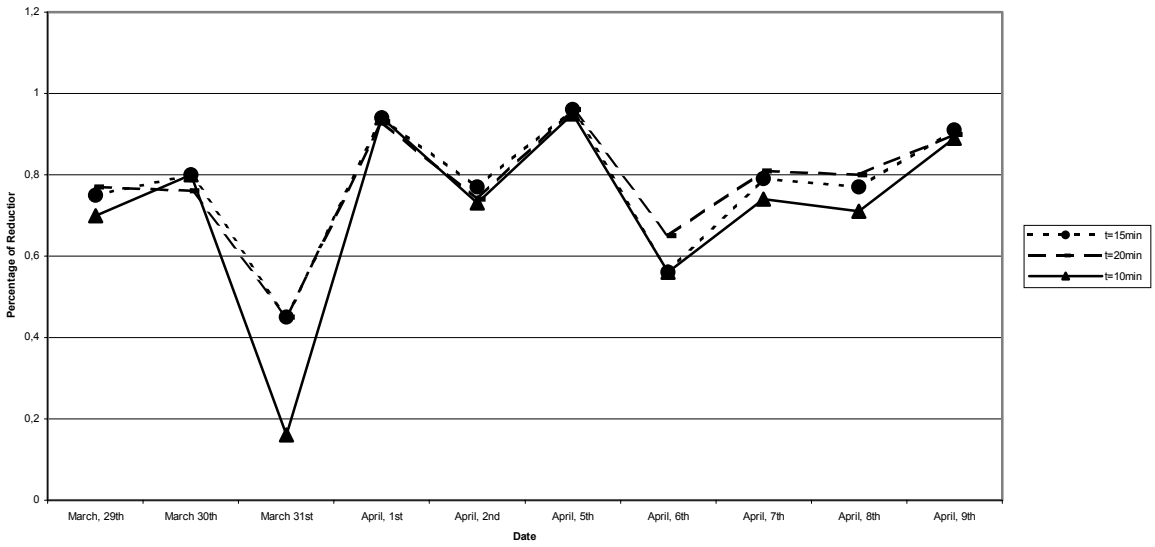


Figure 3. Reduction on the number of alarms detected on the labeled dataset DARPA99 using SOARA.

During experiments with DARPA99 dataset, SOARA was tested using different values of *t* (i.e. time intervals *Ii* of size *t*). SOARA ran with 10, 15 and 20 minute interval. Figure 3 shows the reduction on the number of alarms to be inspected over dataset DARPA 99. Figure 4 shows the effectiveness or percentage of true positives detected on the labeled dataset DARPA99 using SOARA. Most true positives detected were in the subset of raw alarms indicated by t-alarms (i.e., the signal with the biggest error value).

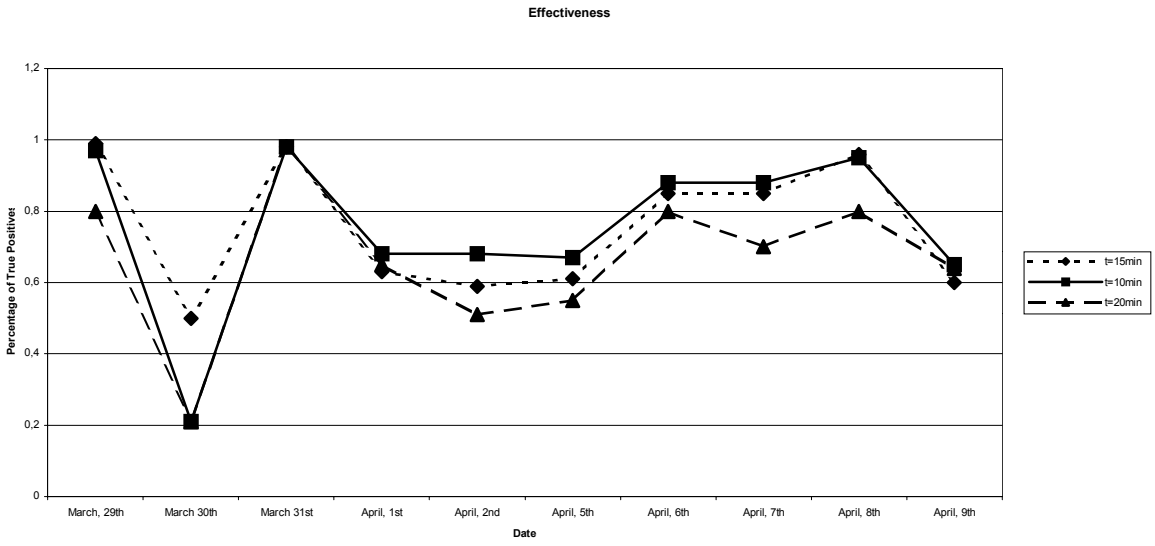


Figure 4. Effectiveness of SOARA (i.e., percentage of true positive alarms detected) on the labeled dataset DARPA99 using SOARA.

5 Conclusion

Experimental studies with SOARA aims at reducing the number of intrusion alarms to be examining by network security experts. Multidimensional signals aggregate alarms by several feature attributes using sketch summaries. A cache memory implementation of sketches allows efficient use of resources. The time-decaying moving median procedure finds exceptional values or changes on the stream of alarms by comparing the observed rate of alarms against the predicted one. Tests on several datasets show promising reduction on the number of alarms to be inspected. In particular, the labeled dataset DARPA99 showed that SOARA could improve the effectiveness of network security operators by focusing on the most interesting data.

In the future, SOARA will integrate additional network data to further reduce false positive alarms. Passive operating system fingerprints data could used to reduce the number of alarms triggered for the wrong operating system. This will reduce the number of t-alarms without significant computational demand. Besides, network flow data could be used to deduce open ports or service provided by distinct hosts on the network. The effectiveness of SOARA could be improved by keeping track of signals associated with services offered on the network only.

Further testing needs to be done on labeled datasets like DARPA99 to verify the reduction on the number of false positives versus the percentage of true positives detected by t-alarms.

In addition, a distributed SOARA system is being designed to exploit the fact that sketch summaries can be exchanged in a distributed IDS environment to improve the efficiency and cooperation between composing IDS.

References

[1] Advanced Network Management Lab, *The Abilene Project*, University of Indiana, Bloomington, Indiana, USA.

[2] Axelsson, Stefan, "The base-rate fallacy and the difficulty of intrusion detection," Journal of the ACM Transactions on Information Systems Security, 3(3), pp. 186-205, 2000.

[3] Babcock, B., Datar, M., Motwani, R. and O'Callaghan, L., "Maintaining variance and k-medians over data stream windows," In Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, 2003, pp. 234-243, San Diego, California.

[4] Barán, B., Aguayo, S., "Caracterización del Backbone ATM de la Red Metropolitana del Sector Publico Paraguayo", XXIV Conferencia Latinoamericana de Informática, Quito, Ecuador 1998.

[5] Barford, Paul, Kline, Jeffery, Plonka, David and Ron, Amos, "A signal analysis of network traffic anomalies," In Proceedings of the second ACM SIGCOMM Workshop on Internet measurment, pp. 71-82, Marseille, France, 2002.

[6] Cohen, Edith and Strauss, Martin, "Maintaining time-decaying stream aggregates," In Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, 2003, pp. 223-233, San Diego, California.

[7] Cuppens, Frederic and Miege, Alexandre, "Alert correlation in a cooperative intrusion detection framework," In Proceedings of the 2002 IEEE Symposium on Security and Privacy, May 2002.

[8] Dong, G., Han, J., Lakshmanan, L. V.S., Pei, J., Wang, H. and Yu, P. S., "Online mining of changes from data streams: Research problems and preliminary results, " In Proceedings of the 2003 ACM SIGMOD Workshop on Management and Processing of Data Streams, San Diego, CA, June 2003.

[9] Erbacher, R. F. and Sobylak, K., "Improving Intrusion Analysis Effectiveness," Workshop on Statistical and Machine Learning Techniques in Computer Intrusion Detection, George Mason University, September 24-26, 2003.

[10] Eskin, E., Arnold, A., Prerau, M., Portnoy, L. and Stolfo, S., "A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data," Data Mining for Security Applications. Kluwer 2002.

[11] Gilbert, A. C., Kotidis, Y., Muthukrishnan, S. and Strauss, M., "Surfing Wavelets on Streams: One-Pass Summaries for Approximate Aggregate Queries," In Proceedings of the 27th International Conference on Very Large Data Bases, pp 79-88, Rome, Italy, 2001.

[12] Hussain, Alefiya, Heidemann, John and Papadopoulos, Christos, "A framework for classifying denial of service attacks," In Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications, pp 99-110, Karlsruhe, Germany, August 25-29, 2003.

[13] Julisch, Klaus, "Mining alarm clusters to improve alarm handling efficiency," In 17th Annual Computer Security Applications Conference (ACSAC), pp 12-21, December 2001.
[14] Julisch, Klaus and Dacier, Marc, "Mining Intrusion Alarms for Actionable Knowledge," in SIGKDD'02, Edmonton, Alberta, Canada, 2002.
[15] Julisch, Klaus, "Clustering Intrusion Detection Alarms to Support Root Cause Analysis", in ACM Transactions on Information and System Security 6(4), November 2003.
[16] Krishnamurthy, B., Sen, S., and Zhang, Y. and Chen, Y., "Sketch-based change detection: methods, evaluation, and applications," In Proceedings of the 2003 ACM SIGCOMM conference on Internet measurement, pp 234-247, Miami Beach, FL, USA, 2003.
[17] Lee, Wenke and Stolfo, Sal. "Data Mining Approaches for Intrusion Detection" In Proceedings of the Seventh USENIX Security Symposium (SECURITY '98), San Antonio, TX, January 1998.
[18] Levera, J., Barán, B., and Grossman R., *"Experimental Studies Using Median Polish Procedures to Reduce Alarm Rates in Data Cubes of Intrusion Data"*, Workshop on Intelligence and Security Informatics for National and Homeland Security, Hsinchun Chen, Reagan Moore, Daniel Zeng, John Jeavitt, editors, LNCS 3073, pages 482-491, Springer Verlag, New York, 2004.
[19] Lincoln Laboratory, Massachussets Institute of Technology, DARPA 99 Intrusion Detection Data Set Attack Documentation. [Online] Available: http://www.ll.mit.edu/IST/ideval/docs/1999/attackDB.html.
[20] Maloof, M. A. and Michalski, R. S., "A Method for Partial-Memory Incremental Learning and its Application to Computer Intrusion Detection," In Proceedings of the 7th International Conference on Tools with Artificial Intelligence, pp 392-397, Washington, DC, November 5-8, 1995.
[21] Manganaris, S., Christensen, M., Zerkle, D., and Hermiz, K., "A Data Mining Analysis of RTID Alarms," Computer Networks, 34(4) , October 2000.
[22] Peng Ning, Yun Cui and Douglas S. Reeves, "Constructing attack scenarios through correlation of intrusion alerts, "In Proceedings of the 9th ACM conference on Computer and communications security, Washington, DC, USA, 2002.
[23] Portnoy, L., Eskin, E. and Stolfo, S. J.. "Intrusion detection with unlabeled data using clustering" In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001). Philadelphia, PA: November 5-8, 2001.
[24] Roesch, M., "Snort - lightweight intrusion detection for networks," In Proceedings of Thirteenth Systems Administration Conference (LISA '99), pp. 229--238, The USENIX Association, Berkeley, California, 1999.
[25] Valdes, Alfonso and Skinner, Keith, "Probabilistic Alert Correlation," Workshop on Recent Advances in Intrusion Detection, Lecture Notes in Computer Science, Number 2212, Springer-Verlag, 2001.
[26] Vert, G., Frincke, D. A., and McConnell, J. C., "A visual mathematical model for intrusion detection" In Proceedings of the 21st National Information Systems Security Conference, Crystal City, Arlington, VA, USA, October 5-8 1998.
[27] Yamanishi, Kenji and Takeuchi, Jun-ichi, "A unifying framework for detecting outliers and change points from non-stationary time series data," In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, pp 676-681, Edmonton, Alberta, Canada, 2002.
[28] Ye, N. and Li, X., "A Scalable Clustering Technique for Intrusion Signature Recognition," In Proceedings of the 2001 IEEE, Workshop on Information Assurance and Security, United States Military Academy, West Point, NY, 5-6 June, 2001.

Qualidade de Serviço com Ganho de Multiplexação Estatística

Sibelius Lellis Vieira
Universidade Católica de Goiás, Departamento de Computação
Goiânia, GO, BRASIL, 74605-010
sibelius@ucg.br

Abstract

The Internet is still largely based on the best effort service, which does not provide enough support for multimedia applications with strict timing requirements, such as Voice over IP and videoconferencing. The network should provide to these applications bounds in the maximum delay and packet rate loss. In order to determine the required network characteristics to provide these services, we use a formal modeling of traffic and bandwidth service based on the network calculus. The network calculus provides a framework to identify the necessary resources to a given application, based on their traffic profile. The backlog and delay bounds can be evaluated given a representation of the service offered by the node and by the network as a whole. In general, the statistical analysis of the quality of service can provide a gain in the resource utilization when compared to the deterministic analysis. We try to identify and compare the deterministic and statistical calculus in this sense.

Keywords: Quality of Service, Network Calculus, Performance Evaluation.

Resumo

O serviço de melhor esforço, disnível em larga escala na Internet, não é suficiente para garantir um suporte adequado para aplicações com requisitos temporais rígidos, tais como Voz sobre IP and videoconferência. Este suporte tem como meta fornecer a estas aplicações garantias de atraso máximo e taxa máxima de perda de pacotes e deve ser estabelecido em termos de gerência de banda, controle de buffers e regulagem de tráfego. Neste trabalho, empregamos uma modelagem formal de controle de tráfego e serviço de banda baseada em cálculo de rede que tem como propriedade a identificação, a partir das características do tráfego, os recursos necessários para assegurar a qualidade das aplicações. O tamanho das filas e limites de atraso podem ser estimados a partir de uma representação do serviço oferecida pelo rede. Em geral, a especificação de qualidade em termos estatísticos pode fornecer um ganho na utilização dos recursos da rede em relação à qualidade determinística. Procuramos identificar e relacionar as vantagens e desvantagens do uso do cálculo de rede estatístico em relação ao cálculo determinístico.

Palavras chaves: Qualidade de Serviço, Cálculo de Rede, Análise de Desempenho de Redes.

1 INTRODUÇÃO

A transmissão de dados entre os elementos que compõe a Internet se baseia no serviço de melhor esforço, que é atualmente o único tipo de serviço oferecido em larga escala nesta rede. O objetivo deste serviço, como o nome sugere, é processar a comunicação dos pacotes de forma otimizada, através do uso racional dos recursos. Contudo, tal serviço não dá garantias específicas de desempenho ou de confiabilidade para as aplicações. Com o crescimento da Internet, este serviço se mostrou fundamental, pois permitiu que os elementos de rede tais como os roteadores se mantivessem simples, sendo amplamente disponibilizado. Além disto, tem propriedades importantes, tais como a escalabilidade e robustez, sendo adequado para grande parte das aplicações tradicionais. Entretanto, a Internet vem sendo utilizada cada vez mais por um contingente de aplicações com requisitos temporais e de consumo de taxa mais rígidos. As novas aplicações, tais como telefonia IP, video-conferência, *streams* de vídeo e áudio e outros necessitam de serviços mais sofisticados do que o serviço de melhor esforço pode oferecer. A telefonia IP, por exemplo, é uma das muitas aplicações que têm evidenciado a necessidade de diversas mudanças na infraestrutura da rede, pois experiências associadas a conversações telefonicas demonstram que um parâmetro tal como o atraso fim-a-fim deve ser limitado a 200ms. O tráfego destas aplicações possui requisitos drasticamente diferentes dos associados à rede de pacotes tradicional, tais como a limitação de atraso e a garantia de largura de banda mínima. Portanto,

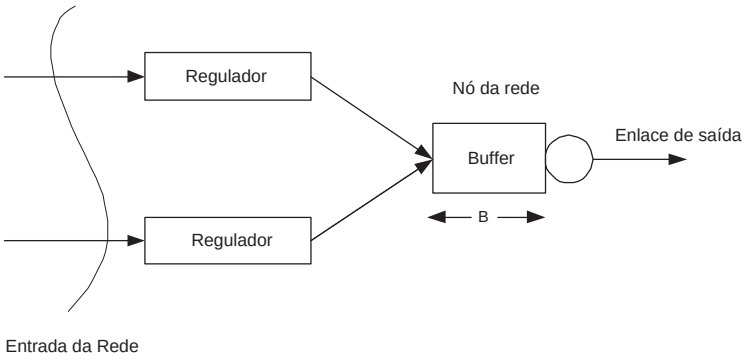


Figure 1: Um elemento de rede ou nó pode ser visto como um conjunto de portas de saída, cada qual com um buffer de tamanho finito. Em geral, múltiplos fluxos de entrada podem ser direcionados e multiplexados em uma única porta de saída. Fluxos entrantes na rede são regulados no primeiro nó através de um envelope de entrada A^*.

faz-se necessário analisar, modelar, desenvolver e implementar novos protocolos e serviços que permitam o suporte de aplicações com características de comunicação multimídia [11].

As novas aplicações requisitam uma rede tenha a capacidade de dar o suporte adequado a uma grande variedade de tráfego com requisitos de desempenho diferenciados e críticos. Esta infraestrutura de comunicação deve conter mecanismos de controle e gerenciamento de tráfego, assim como um controle do comportamento do tráfego submetido por diversas aplicações [12]. A análise de problemas de controle de tráfego em redes, escalonamento de pacotes e encaminhamento é extremamente importante para a garantia da qualidade de serviço (QoS), exemplificada anteriormente como o atraso fim-a-fim. De forma mais geral, esta QoS é especificada em termos de atraso máximo e perda de pacotes. O controle de tráfego assegura que os recursos da rede sejam divididos de forma eficiente, de tal forma a maximizar o uso destes recursos enquanto assegura o cumprimento da QoS para os usuários. Para assegurar o escalonamento de recursos da rede de forma a atender aos requisitos de QoS, é necessário também que cada fonte caracterize seu próprio tráfego. O gerenciamento de tráfego na rede é importante para assegurar que a qualidade de serviço negociada para as sessões seja mantida.

Neste contexto, os fluxos devem estabelecer contratos com a rede de maneira a limitar, em algum sentido, a quantidade de tráfego que os mesmos irão injetar na rede em determinado intervalo de tempo. O estabelecimento e manutenção destes contratos são pontos importantes para a garantia de fornecimento de serviço por parte da rede. Reguladores do tipo balde furado, por exemplo, são mecanismos convenientes para a definição e garantia de cumprimento dos contratos de tráfego [3]. As fontes que apresentam conformidade com a especificação de tráfego são ditas fontes reguladas por envelopes de tráfego.

É fundamental se determinar a quantidade de recursos necessária para garantir a QoS, expressada em termos de atrasos máximos ou perdas de pacotes. O atualmente denominado cálculo de rede oferece uma série de métodos para que esta quantidade seja estabelecida [4, 5]. Através deste formalismo, é possível calcular, usando como entrada as necessidades das aplicações e sua caracterização de tráfego, os recursos necessários para a garantia de limites nos valores de atraso e de perda de pacotes. O cálculo de rede determinístico provê uma abordagem formal elegante para análise de pior caso, que pode ser usada para derivar limites máximos em atrasos e tamanho da fila para uma grande variedade de mecanismos de escalonamento [2]. A alocação de recursos é feita através do conceito de curva de serviço, que aloca para cada conexão uma quantidade de recursos em pior caso. Uma vantagem do cálculo de rede determinístico é a sua capacidade de determinar os atrasos máximos e tamanho de filas em vários nós da rede.

Pode se argumentar, contudo, que a garantia absoluta de que nenhum pacote será perdido ou sofrerá um atraso maior do que o estabelecido é por demais conservadora para aplicações de mídia contínua, as quais podem tipicamente tolerar uma taxa pequena de perdas. De fato, os usuários não percebem qualquer degradação de qualidade quando a perda de pacotes é pequena e pouco frequente, especialmente se o receptor emprega técnicas de cancelamento de erros. Além do mais, esquemas que garantem a entrega total dos pacotes tipicamente tem uma baixa capacidade de transmitir tráfego com características de rajada. Posto de outra forma, as abordagens determinísticas que garantem que não há perda requerem uma superestimação dos recursos de rede necessários para garantir a QoS [1]. Portanto, a utilização de modelos determinísticos leva a um uso ineficiente dos recursos de rede.

Estas características do cálculo determinístico levantam questões importantes. A primeira é a possibilidade de desenvolver uma abordagem que provê garantias estatísticas de QoS em uma rede, ou seja, um limite na fração de tráfego que exceda um parâmetro de QoS tal como o atraso máximo [7]. A garantia estatística

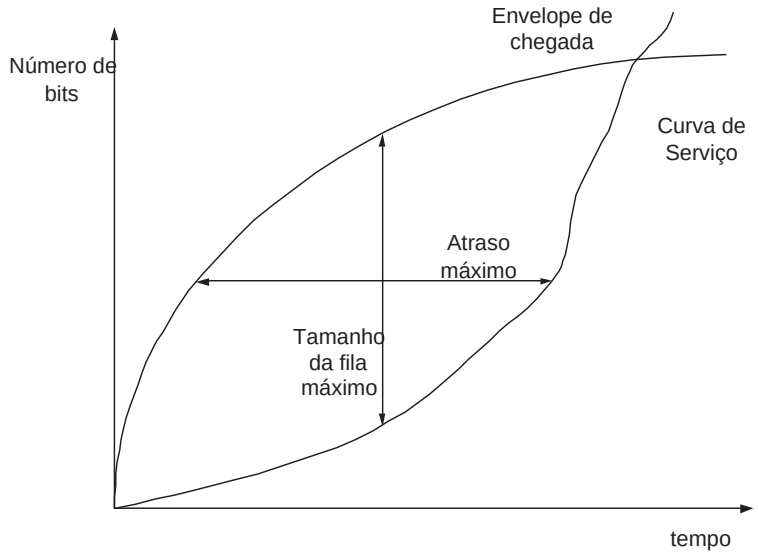


Figure 2: Illustração da relação entre envelope, curva de serviço, distância horizontal e distância vertical. A diferença horizontal indica o atraso máximo que pode ser associado ao fluxo e a diferença vertical, o tamanho máximo de ocupação de buffers.

de QoS em um contexto de rede é notoriamente difícil, pois os fluxos de tráfego normalmente perdem suas características estatísticas originais no processo de compartilhamento de filas de saída. A segunda questão diz respeito ao ganho advindo desta multiplexação estatística em relação à abordagem determinística [6].

Uma característica das redes de pacotes é a sua habilidade de explorar a multiplexação estatística de fontes de tráfego e portanto, alcançar um alto grau de utilização dos recursos disponíveis. Este ganho em termos de multiplexação estatística pode ser explorado tirando vantagem das propriedades estatísticas, da independência das fontes e dos diferentes requisitos de QoS das mesmas [10]. Os modelos estatísticos, baseados em um cálculo estatístico de rede, podem estabelecer garantias da forma probabilística, tais como a probabilidade de um valor de atraso exceder um limite pré-determinado é de $10^{-\epsilon}$, onde ϵ varia entre 3 e 9. Desta forma, uma pequena fração de tráfego pode violar as especificações de QoS. Estas abordagens estatísticas podem permitir um ganho substancial no uso dos recursos, evidenciado em uma relação que expressa que os recursos necessários para garantir estatisticamente N fluxos são bem menores do que os recursos para garantir deterministicamente estes mesmos fluxos.

Este trabalho procura investigar um importante aspecto do cálculo de rede determinístico e estatístico, qual seja o nível de utilização de recursos indicado pelo número de fluxos admissíveis em um nó. O esquema de gerenciamento de tráfego tem os seguintes componentes: cada fluxo deve ser regulado na entrada da rede por uma elemento que implemente um envelope de tráfego, todos os nós empregam um mecanismo de multiplexação de tráfego com memória e o controle de admissão é baseado na suposição de que o tráfego de uma fonte é independente do tráfego de outra na entrada da rede.

Na seção II, descrevemos as bases do cálculo determinístico e apresentamos os principais resultados que nos permitem obter os parâmetros de interesse dos fluxos, tais como limites de atraso, tamanho de filas e funções de limitação de tráfego. Na seção III, apresentamos a extensão do cálculo determinístico para obter garantias estatísticas e discutimos os resultados do cálculo estatístico e seus limites. Na seção IV, apresentamos uma análise de comparação de desempenho entre o cálculo determinístico e o cálculo estatístico para situações de fluxos fim-a-fim. Finalmente, na seção V apresentamos as conclusões.

2 CÁLCULO DETERMINÍSTICO

Nesta seção, apresentamos algumas definições e resultados que serão considerados e descritos coletivamente como "cálculo de rede". O princípio geral do cálculo de rede é fornecer uma abordagem formal para a obtenção de parâmetros importantes para caracterizar o desempenho de uma rede, tais como o atraso máximo ou a taxa de perda de pacotes. Através de uma coleção de resultados oriundos de uma álgebra conhecida como *min-plus*, podemos caracterizar sistemas de filas utilizados em redes de comunicação. Como exemplo, podemos utilizá-lo para entender os cálculos de atraso usados pelos servios garantidos propostos pelo IETF, como modelo comum para escalonadores, para identificar a influência de reguladores sobre o atraso, para calcular a banda efetiva necessária de um fluxo com garantias de atraso, entre outras aplicações.

Vamos considerar como exemplo funções não-decrescentes no tempo. A função $f(\ )$ é não-decrescente quando $f(s) \leq f(t)$ sempre que $s \leq t$. Para duas funções não-decrescentes f_1 e f_2, denominamos a operação de convolução *min-plus* entre f_1 e f_2 como $f_1 \otimes f_2$ correspondente à operação de convolução padrão e definida da seguinte forma:

$$f_1 \otimes f_2(t) = inf_{0 \leq u \leq t}\{f_1(u) + f_2(t-u)\} \quad \forall t \geq 0 \tag{1}$$

Seja um fluxo de dados descrito por sua função de chegada $A(t)$, que é igual ao número de bits do fluxo que chega em um nó da rede em um intervalo de tempo $[0,t]$. Portanto, a função $A(t)$ representa uma quantidade aleatória, cuja distribuição não é conhecida. Como o tráfego é cumulativo, podemos considerar $A(t)$ como função não-decrescente. Dada uma função não-decrescente A^*, dizemos que o fluxo é restrito por A^* se e somente se para todo $s \leq t$, temos que $A(t) - A(s) \leq A^*(t-s)$. Isto é equivalente a afirmar que $A(t) \leq A \otimes A^*(t)$ para qualquer $t \geq 0$.

A função A^* é chamada de curva de chegada ou envelope de fluxo. Por exemplo, um fluxo controlado por um balde furado tem uma curva de chegada na forma $A^*(t) = \sigma + \rho t$. A função A^* pode ser qualquer função não-negativa e não-decrescente, mas para definir uma restrição consistente deve também ser sub-aditiva, o que implica em $A^*(s+t) \leq A^*(s) + A^*(t)$ para qualquer $s, t \geq 0$. Na Figura 1, ilustramos uma situação em que cada fonte tem um envelope definido por um regulador. A utilidade básica do envelope do fluxo é garantir que os fluxos que entram na rede tem um tráfego máximo definido e controlado pelos reguladores que implementam este envelope.

Consideremos $\mathcal{S}$ um sistema visto na forma de uma caixa-preta, ou seja, este sistema é caracterizado apenas pela sua chegada e pela sua saída. Este sistema também pode ser representado graficamente pela ilustração da Figura 1. O sistema $\mathcal{S}$ recebe dados de entrada e os retransmite na saída após um atraso variável. Seja $A(t)$ a função de entrada, representando o tráfego de chegada e $D(t)$ a função de saída, esta última indicando o número de bits que deixam o sistema no intervalo de tempo $[0,t]$. A fila no tempo t tem um tamanho igual a $A(t) - D(t)$ e representa o número de bits dentro do sistema, supondo o sistema vazio no tempo 0. De forma similar, o atraso virtual no tempo t é dado por:

$$d(t) = inf\{T : T \geq 0 \ \ e \ \ A(t) \leq D(t+T)\} \quad \forall t \geq 0 \tag{2}$$

Este é o atraso que seria associado a um bit chegando no tempo t se todos os bits que foram recebidos antes deste forem servidos na ordem de chegada. Se a função de saída for contínua, então $D(t+d(t)) = A(t)$.

Os resultados do cálculo de rede fornecem regras computacionais para o limite de atrasos virtuais e tamanho de fila para sistemas arbitrários que representam as redes de dados. Dizemos que o sistema $\mathcal{S}$ oferece uma curva de serviço S se e somente se:

$$D(t) \geq A \otimes S(t) \quad \forall t \geq 0 \tag{3}$$

Na prática, isto é equivalente a afirmar que para todo $t \geq 0$, existe algum $t_0 \geq 0$, com $t_0 \leq t$ tal que $A(t) - D(t_0) \geq S(t - t_0)$. Por exemplo, em um sistema de escalonamento no qual uma parte da banda é reservada para um fluxo com taxa garantida r baseado em modelo de fluidos, o sistema oferece ao fluxo uma curva de serviço $S(t) = rt$. O conceito de curva de serviço pode ser relacionado ao de escalonamento, pois para cada tipo de escalonamento, podemos derivar uma curva de serviço representando este escalonamento. Porém, as curvas de serviço são mais gerais, e em alguns casos não representam disciplinas de escalonamento factíveis. Como exemplo de uma disciplina não-factível, temos a curva de serviço $S(t) = 0$ para $t < d$ e $S(t) = \infty$ para $t \geq d$.

Uma série de resultados foram derivados para o atraso e o tamanho de fila em um nó da rede, bem como para um caminho constituído de vários nós, dados a função de entrada e a curva de serviço [2]. O primeiro supõe a existência de um fluxo, com envelope A^* que atravessa um sistema que oferece uma curva de serviço S. O tamanho da fila $A(t) - D(t)$ para todo t é limitado por:

$$A(t) - D(t) \leq sup_{s \geq 0}\{A^*(s) - S(s)\} \quad \forall t \geq 0 \tag{4}$$

Se o sistema é um buffer simples em um nó, este tamanho pode ser interpretado como o tamanho instantâneo da fila. Por outro lado, se o sistema é mais complexo, tal como um conjunto de buffers em nós de um caminho, então este tamanho é o número de bits "em trânsito", supondo que podemos observar tanto a entrada quanto a saída simultaneamente. Este resultado indica que o tamanho da fila é limitado pela diferença vertical entre o envelope e a curva de serviço, conforme ilustrado na Figura 2.

Para descrevermos o próximo resultado, introduzimos uma nova operação, que pode ser interpretada como uma operação inversa à da convolução. Dizemos que a função $f_1 \oslash f_2(t) = sup_{u \geq 0}\{f(t+u) - f(u)\}$ é a deconvolução das funções f_1 e f_2. Supondo a existência de um fluxo com características dadas anteriormente,

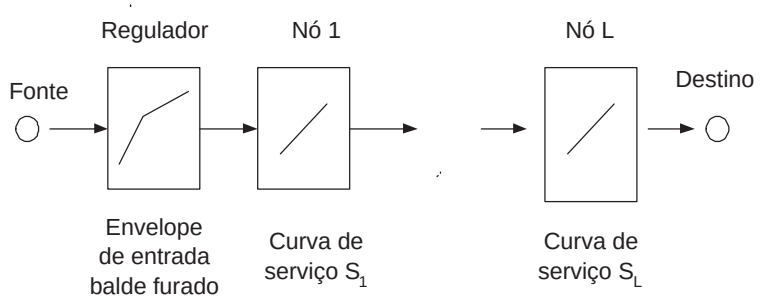


Figure 3: Um fluxo criado em uma fonte atravessa uma rede através de um caminho com N nós, cada qual oferecendo uma curva de serviço. Na entrada da rede, um regulador garante a conformidade com o envelope de entrada.

o fluxo de saída é restringido por um envelope dado por $A^* \oslash S(t)$. Portanto, dado um envelope de entrada e curvas de serviço, podemos obter envelopes para o tráfego em qualquer nó da rede.

Para duas funções não-decrescentes α e β, definimos a diferença horizontal entre as duas curvas como sendo $h(\alpha,\beta) = sup_{s\geq 0}(inf\{T : T \geq 0 \quad e \quad \alpha(s) \leq \beta(s+T)\})$. A definição pode ser entendida intuitivamente da seguinte forma: se α e β são contínuas e estritamente crescentes, então para todo t existe no máximo um número $d(t)$ tal que $\alpha(t) = \beta(t + d(t))$. Neste caso, $h(\alpha,\beta) = sup_t d(t)$. Em outras palavras, $h(\alpha,\beta)$ não é nada mais do que a fórmula usada para a computação do atraso máximo, que pode ser reescrita como $d(t) \leq h(\alpha,\beta)$ para qualquer t. O valor do atraso máximo pode ser representado como a diferença horizontal entre o envelope e a curva de serviço, conforme ilustrado na Figura 2. Desta forma, podemos computar o atraso máximo d como dado abaixo:

$$d = sup_{t\geq 0} inf\{T : T \geq 0 \quad e \quad A(t-T) \leq S(t)\} \tag{5}$$

Estes resultados podem ser aplicados em um caminho de rede envolvendo vários nós. Suponha a existência de dois sistemas $\mathcal{S}$ e $\mathcal{F}$ com curvas de serviço S e F, respectivamente. A concatenação destes dois sitemas oferece uma curva de serviço $T = S \otimes F$. Por exemplo, na Figura 3 ilustramos um fluxo com um regulador de entrada implementando um envelope de tráfego para este fluxo e a seguir, uma série de nós que fazem parte do caminho deste fluxo na rede. A curva de serviço de rede para este fluxo pode ser obtida como uma convolução das várias curvas de serviço individuais para o fluxo em cada um dos nós. Outras características do fluxo podem ser obtidas através do envelope de entrada e desta curva de serviço de rede, tais como o atraso fim-a-fim sofrido pelos pacotes deste fluxo.

Um regulador, com curva de regulagem γ é um dispositivo que força uma saída na forma de um envelope com função γ. O regulador atrasa o bit em um buffer sempre que o envio deste bit possa violar as restrições do tráfego de saída. Um balde furado é um regulador cuja função γ tem a forma $\gamma = \sigma + \rho t$, para $t > 0$ e $\gamma = 0$ para $t = 0$. O regulador tem duas propriedades importantes: em primeiro lugar, o regulador não aumenta o limite do atraso. Suponha que um fluxo com envelope de entrada A^* é direcionado para o sistema formado por $\mathcal{S}$ e $\mathcal{F}$ em sequencia. Suponha que um regulador $\gamma \leq A^*$ é inserido entre $\mathcal{S}$ e $\mathcal{F}$. O limite de retardo dado anteriormente é também válido para o sistema com o regulador. O regulador também conserva as propriedades originais do fluxo. Considere um fluxo, com envelope de entrada A^* direcionado para um regulador com curva γ. A saída do regulador ainda é restringida pela curva de entrada original A^*.

3 CÁLCULO ESTATÍSTICO

O cálculo estatístico extende o cálculo determinístico apresentado dentro de um contexto probabilístico no intuito de explorar o ganho de multiplexação estatística. Este ganho de multiplexação estatística pode ser observado concretamente através de mecanismos de controle de admissão, que indicam o número de fluxos aceitáveis em um sistema ou em algum nó. Um dos primeiros métodos que foram estudados extensivamente, o cálculo de banda efetiva, foi motivado por funções que apareceram na teoria de desvios grandes. Expressões para a banda efetiva de um fluxo foram derivados, associando de forma estatística a banda que um fluxo iria necessitar para garantia de sua QoS com uma probabilidade dada. Para esta computação, o utiliza-se uma expressão baseada na função de chegada $A(t)$. Neste trabaho, não investigaremos as relações entre o cálculo estatístico e o cálculo de banda efetiva. Contudo, estas relações tem sido analisadas, e os resultados indicam que o cálculo estatístico pode fornecer uma abordagem mais geral com garantias de desempenho similares ao cálculo de banda efetiva [8].

No formalismo de cálculo estatístico, consideramos as chegadas e partidas em um intervalo de tempo $[0,t]$ vistas como processos estocásticos que satisfazem certas propriedades, e as funções de chegada $A(t)$ e saída $D(t)$ representam estes processos. Para assegurar a validade do cálculo estatístico, precisamos estabelecer certas condições em relação ao processo de chegada:

(A1) Aditividade: Para qualquer $t_1 < t_2 < t_3$, temos que $A(t_1,t_3) = A(t_1,t_2) + A(t_2,t_3)$.

(A2) Sub-aditividade: O envelope $A^*(t)$ é subaditivo.

(A3) Estacionaridade: A função $A(t)$ é estacionária, ou seja, $Pr[A(t,t+\tau) \leq x] = Pr[A(t_1,t_1+\tau) \leq x]$, para qualquer $t, t_1 \geq 0$.

(A4) Independência: As chegadas A_i e A_j para dois fluxos diferentes são estocasticamente independentes.

Estas suposições são feitas apenas na entrada da rede, quando o tráfego está chegando no primeiro nó de seu caminho. Nenhuma suposição adicional é feita com relação ao tráfego dentro da rede. A subaditividade garante que o limite $lim_{\tau\to\infty} A^*(\tau)/\tau$ existe e é denotado por ρ. Desta forma, existe um limite superior para a taxa de chegada para $A(t)$ a longo prazo. A estacionaridade tem como características interessantes o fato de que valores esperados podem ser computados como médias a longo prazo. A independência dos fluxos permite explorar os ganhos com a multiplexação estatística.

3.1 Extensões ao cálculo determinístico

Vamos definir as funções do cálculo estatístico que generalizam os conceitos de envelope e curva de serviço apresentados para o cálculo determinístico. Para os fluxos de entrada, definimos o envelope efetivo $G^\epsilon(\tau)$ para um processo de chegada $A(t)$ representado o tráfego entrante em $[0,t]$ como sendo:

$$Pr[A(t+\tau) - A(t) \leq G^\epsilon(\tau)] \geq 1-\epsilon \quad \forall t,\tau \geq 0 \tag{6}$$

De forma simplificada, o envelope efetivo representa um limite estacionário para os processos de chegada. De acordo com esta definição, é possível que o tráfego de chegada exceda a quantidade permitida pelo envelope efetivo com uma pequena probabilidade. Conforme será descrito posteriormente, é possível obter limites para a função de envelope efetivo a partir da função que representa o envelope de chegada. Estes limites são apertados, representando de forma acurada a função do envelope efetivo.

De forma análoga, podemos definir a curva de serviço efetiva como sendo uma medida da probabilidade do serviço disponível para o fluxo. A curva de serviço efetiva garante que uma grande quantidade de tráfego de entrada estará disponível na saída, através da função $D(t)$, como resultado do serviço efetivo. Dado um processo de chegada $A(t)$, a curva de serviço efetiva S^ϵ é uma função não-negativa que satisfaz para todo $t \geq 0$

$$Pr[D(t) \geq A \otimes S^\epsilon(t)] \geq 1-\epsilon \tag{7}$$

A definição da curva de serviço efetiva impõe uma dificuldade para estabelecer uma curva de serviço global. Isto pode ser entendido se observarmos que, como a definição da curva de serviço efetiva não assegura que o valor de $D(t)$ não é menor do que o $inf_{s\geq t}\{A(t-s) + S^\epsilon(s)\}$, não é possível garantir uma expressão simples para a concatenação das curvas de serviço. Uma forma de diminuir a dificuldade no cálculo desta concatenação é adicionar uma suposição relativa ao escopo do $infimum$. Esta suposição se baseia na existência de um valor T tal que

$$Pr[D(t) \geq inf_{s\leq T}\{A(t-s) + S^\epsilon(s)\}] \geq 1-\epsilon \quad \forall t \geq 0 \tag{8}$$

Portanto, T limita o escopo da convolução. Em geral, o valor de T está relacionado a um intervalo de tempo no qual a convolução faz sentido. Por exemplo, em uma porta de saída de um nó da rede, os intervalos de tempo importantes são aqueles em que existe tráfego requisitando serviço nesta porta, ou seja, os intervalos nos quais o buffer não está vazio. Os períodos nos quais o buffer está vazio tem solução trivial, pois não existe demanda de tráfego. O tamanho destes intervalos ocupados pode ser relacionado ao valor de T. Por exemplo, para curvas de serviço na forma $S(t) = kt$, o cálculo determinístico fornece um limite superior para o valor de T na dado como [8]:

$$T = inf\{\tau > 0 : A^*(\tau) \leq S(\tau)\} \tag{9}$$

O valor de T dado pela eq. (9) fornece um limite superior conservador, no sentido de que pode ser usado para o cálculo estatístico, que fornece limites mais otimistas [8].

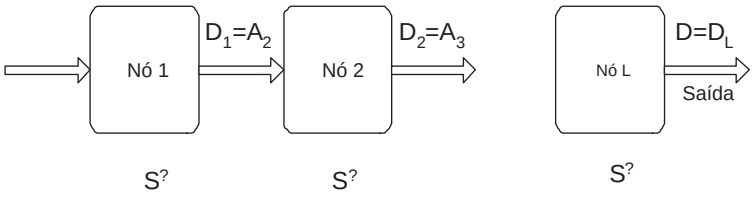


Figure 4: Nós do caminho de um fluxo com curva de serviço S^{ϵ}. A saída de um nó D_i corresponden a entrada do próximo nó A_{i+1}.

Os resultados apresentados no cálculo determinístico podem ser aplicados para o cálculo estatístico, de acordo com as proposição abaixo:

Suponha que G^{ϵ_g} é um envelope efetivo para as chegadas A em um nó e que S^{ϵ_s} seja uma curva de serviço efetiva satisfazendo a eq. (8) para algum $T < \infty$. Defina ϵ como $\epsilon = \epsilon_s + T\epsilon_g$. Temos que:

1 Envelope de saída: A função $G^{\epsilon_g} \oslash S^{\epsilon_s}$ é um envelope efetivo para o tráfego de saída do nó.

2 Limite de tamanho de fila: A função $G^{\epsilon_g} \oslash S\epsilon_s(0)$ é um limite probabilístico para o tamanho da fila, no sentido de que, para qualquer $t \geq 0$, temos $Pr[B(t) \leq G^{\epsilon_g} \oslash S^{\epsilon_s}(0)] \geq 1 - \epsilon$.

3 Se $d > 0$ satisfaz $sup_{\tau \leq T}\{G^{\epsilon_g}(\tau - d) - S^{\epsilon_s}(\tau)\} \leq 0$, então d é um limite de probabilístico do atraso, no sentido de que, para qualquer $t \geq 0$, temos $Pr[W(t) \leq d] \geq 1 - \epsilon$.

Podemos verificar que este teorema generaliza os resultados do cálculo determinístico, que ocorrem no limite de $\epsilon_g, \epsilon_s \to 0$.

O cálculo estatístico fornece também uma expressão probabilística para a convolução de curvas de serviço de vários elementos de rede dispostos em sequencia. Esta expressão fornece o serviço dado pela rede como um todo em um determinado caminho. Considere tal caminho de um fluxo como ilustrado na Figura 4 . Em cada nó, a chegada tem alocada uma curva de serviço, indicada como S^{i,ϵ_s} para o nó i. De forma similar à eq. (8), supomos que cada nó satisfaz a expressão $Pr[D^i(t) \geq inf_{\tau \leq T^i}\{A^i(t-\tau) + S^{i,\epsilon_s}(\tau)\}] \geq 1 - \epsilon_s$, para $T_i < \infty$, com $i = 1, \ldots, L$. Podemos obter uma curva de serviço efetiva da rede, $S^{tot,\epsilon}$ dada como:

$$S^{tot,\epsilon} = S^{1,\epsilon_s} \otimes S^{2,\epsilon_s} \otimes \ldots \otimes S^{L,\epsilon_s} \tag{10}$$

A violação de probabilidade é dada por $\epsilon = \epsilon_s \sum_{i=1}^{L}(1 + (i-1)T^i)$. Este valor de ϵ se degrada a medida que o fluxo atravessa os vários nós do caminho, tornando limitada a abordagem estatística baseada em envelope efetivo e curva de serviço efetiva.

3.2 Multiplexação Estatística

A partir da definição do envelope efetivo G^{ϵ} dada na eq. (6), podemos obter o valor do envelope efetivo baseado em funções de geração de momento das distribuições dos processos de chegada $A(t)$. Sabemos que a função de geração de momento da distribuição de $A(t)$ pode ser dada por $M(s,t) = E[e^{A(\tau,\tau+t)s}]$ [9]. A estacionaridade garante que as funções $M(s,t)$ não dependem de t. Através da utilização dos limite de Chernoff, que garante um limite superior para $P[Y \geq y]$, onde Y é uma variável aleatória e y é um valor possível para esta variável, através da expressão $P[Y \geq y] \leq e^{-sy}E[e^{A(\tau,\tau+t)s}]$ [1], temos que:

$$Pr[A \geq Nx] \leq [e^{-xs}(1 + \frac{\rho\tau}{A^*(\tau)}(e^{sA^*(\tau)} - 1)]^N \tag{11}$$

onde N é o número de fluxos agregados pelo mesmo envelope efetivo. Utilizando a definição de envelope efetivo, obtemos que o valor do envelope efetivo é dado por $G^{\epsilon}(t) = Nmin(x, A^*(\tau))$, onde x é o menor número satisfazendo

$$(\frac{\rho\tau}{x})^{(x/A^*(\tau))}(\frac{A^*(\tau) - \rho\tau}{A^*(\tau) - x})^{1-(x/A^*(\tau))} \leq \epsilon^{1/N} \tag{12}$$

Desta forma, podemos calcular o envelope efetivo de qualquer fluxos ou agregado de fluxos chegando em um nó. Para o próximo nó, podemos utilizar o resultado do Teorema 1 que fornece um envelope de saída efetivo dado por $G^{\epsilon_g} \oslash S^{\epsilon_s}$, onde S^{ϵ_s} é a curva de serviço efetiva do nó.

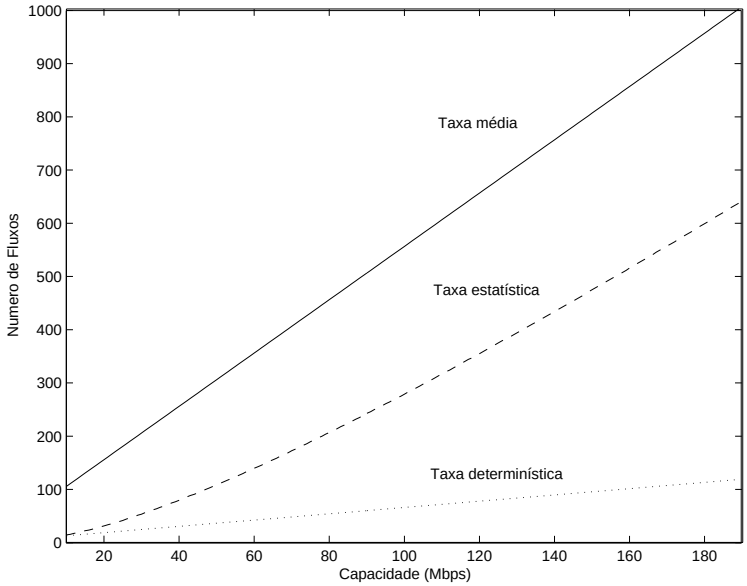


Figure 5: Número de fluxos em função da capacidade do enlace de saída para fluxos do tipo A.

4 AVALIAÇÃO NUMÉRICA

Nesta seção, apresentamos exemplos numéricos para a utilização das saídas dos nós, comparando os envelopes determinísticos com os envelopes efetivos, em uma rede com fluxos com caminhos envolvendo múltiplos nós. Vamos supor que os fluxos sejam individualmente controlados na entrada da rede por um envelope $A^*(\tau) = min\{P\tau, \sigma + \rho\tau\}$, onde P é a taxa de pico do fluxo, ρ é a taxa média e σ é a rajada de dados. Para os nossos experimentos, vamos supor a existência de dois tipos de fluxos: o primeiro, do tipo A, com grande variação entre a taxa média e taxa de pico e com alta quantidade de rajada, tendo como parâmetros $P_A = 2.0Mbps$, $\rho_A = 0.2Mbps$ e $\sigma_A = 0.1Mbit$; o segundo, com pequena variação entre a taxa média e a taxa de pico e com baixa quantidade de rajada, tendo como parâmetros $P_B = 0.2Mbps$, $\rho_B = 0.1Mbps$ e $\sigma_B = 0.01Mbit$.

No primeiro experimento, determinamos a utilização dos nós de saída em relação à capacidade do enlace de saída para o cálculo determinístico. Esta utilização é dada pelo número máximo de fluxos que podem ser acomodados em um enlace de saída sem que a QoS dos fluxos seja comprometida. Como parâmetros de QoS, vamos utilizar o atraso máximo d para os fluxos com um valor de $d = 10ms$. Vamos analisar o comportamento da rede separadamente para os fluxos do tipo A e B. A curva de serviço para o agregado de fluxos do tipo A ou B é dada por $S_R(t) = Ct$, onde C é a capacidade do enlace em Mbps e R representa o agregado. De acordo com os resultados do cálculo determinístico para a diferença horizontal vistos anteriormente, a curva de serviço para cada fluxo separadamente deve obeceder a relação indicada na eq. (5), $A^*(t-d) \leq S(t)$, onde $S^{tot}(t)$ é a curva de serviço de rede para cada fluxo, dada por $S^{tot}(t) = S^1 \otimes \ldots \otimes S^L(t)$, onde L é o número de nós no caminho dos fluxos. Supondo que cada fluxo tem uma curva de serviço por nó dada por $S^i(t) = ct$, onde c é a capacidade necessária para garantir o atraso máximo d, obtemos $S^{tot}(t) = ct$ como resultado das operações de convolução. Desta forma, podemos obter o valor da capacidade equivalente que cada fluxo requisita em cada nó para garantir o atraso máximo d.

Nas Figuras 5 e 6 ilustramos o número de fluxos que podem ser acomodados por enlace para o valor de atraso máximo requisitado. A capacidade máxima para fluxos do tipo A é de $c_A = 1.695$ e obecede a relação $P_A \geq c_a \geq \rho_A$. Para os fluxos do tipo B, $c_B = 0.181Mbps$ e também obedece uma relação equivalente. Na Figura 5, o número de fluxos é dado por uma relação linear na forma $N_A = C/c_A$ e na Figura 6, a mesma relação é dada por $N_B = C/c_B$. Ilustramos também o número de fluxos que poderiam ser acomodados se cada fluxo tivesse garantido apenas a capacidade média do envelope ρ.

No segundo experimento, determinamos a mesma utilização para a situação em que o atraso máximo é garantido estatisticamente. Para tal, calculamos o envelope efetivo dos fluxos do tipo A e B através da solução da eq. (12) para x de tal forma que $d > 0$ satisfaça a relação $sup_{\tau \leq T}\{G^{\epsilon_g}(\tau - d) - S^{\epsilon_s}(\tau)\} \leq 0$, onde $S^{\epsilon_s}(\tau) = Ct$ para o agregado de fluxos. Escolhemos para ϵ_s e ϵ_g o valor de -9. O valor de T_i é derivado da relação que fornece o um limite para o valor do período ocupado para um agregado de fluxos no caso determinístico, conforme apresentado através da eq. (8). Para a curva de serviço e envelope dados, $T_i = 82ms$ para os fluxos do tipo A e $T_i = 122ms$ para os fluxos do tipo B.

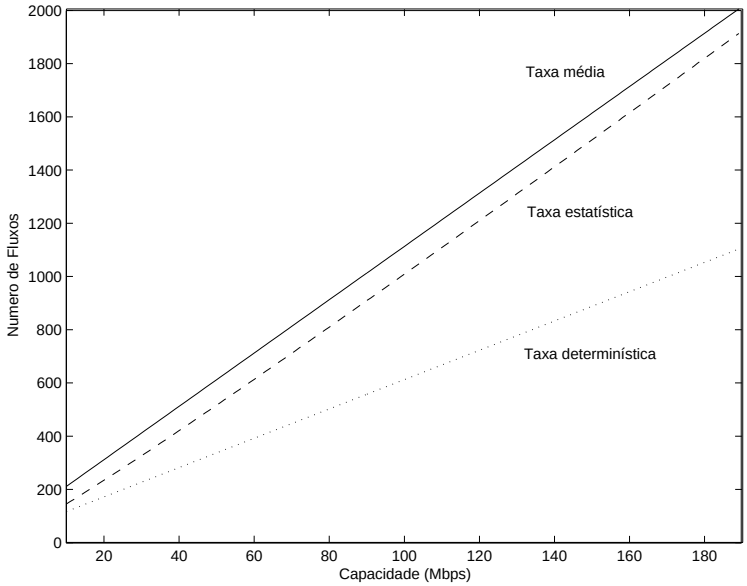


Figure 6: Número de fluxos em função da capacidade do enlace de saída para fluxos do tipo B

Table 1: Degradação do valor de ϵ para caminhos com 5 e 10 nós

Tipo de fluxo	L = 5	L = 10
A	8.10^{-6}	8.10^{-5}
B	10^{-5}	10^{-4}

Nas Figuras 5 e 6 ilustramos novamente o número de fluxos que podem ser acomodados no enlace de para uma garantia estatística do valor máximo de atraso. Podemos observar que o número de fluxos é significativamente maior do que o obtido para o caso determinístico. Na verdade, este número de aproxima do número de fluxos dado utilizando o valor da taxa média como sendo a capacidade reservada para cada fluxo. De forma intuitiva, a capacidade total dividida pelo número de fluxos indica a capacidade equivalente que cada fluxo deve reservar no nó de saída de forma a garantir seus requisitos de atraso máximo. Para a taxa média, esta capacidade reservada por fluxo garante apenas um valor finito para o atraso máximo que o tráfego do fluxo pode sofrer, mas não garante o atraso máximo requisitado. Para a taxa determinística, esta capacidade garante que o fluxo nunca sofre um atraso maior do que d e para a taxa estatística, esta capacidade garante que o fluxo pode sofrer um atraso maior do que d com uma probabilidade muito pequena. Podemos observar que para fluxos com menor rajada, a taxa estatística é mais próxima da taxa média do que para fluxos com maior rajada.

O problema da abordagem estatıstica surge em função da degradação do valor de ϵ. Embora os valores de ϵ_s e ϵ_g sejam muito pequenos, o valor de ϵ se degenera rapidamente quando os fluxos atravessam vários nós no caminho. Na tabela 1, ilustramos os valores de ϵ para $L = 5$ e $L = 10$.

5 CONCLUSÃO

Neste trabalho, procuramos comparar os resultados obtidos pelo cálculo de rede determinístico e estatístico, através do cálculo do número de fluxos que podem ser admitidos em um enlace com determinada capacidade em uma rede com múltiplos nós. Podemos observar a que o número de fluxos empregando a abordagem estatística é sensivelmente maior do que o número de fluxos com garantias determinísticas de atraso. Entretanto, a taxa de perda aumenta a medida que o fluxo atravessa os vários nós do caminho, a ponto do mesmo tornar-se inviável para aplicações com requisitos mais rigorosos em termos de perdas de pacote.

6 Agradecimentos

Agradecemos à PROPE/UCG e à CAPES pelo apoio recebido.

References

[1] R. Boorstyn, A. Burchard, J. Liebeherr and Oottamakorn, C, Statistical service assurances for traffic scheduling algorithms. *IEEE Journal on Selected Areas in Communications*, 18(12):2651–2664, 2000.

[2] J.-Y. Boudec and P. Thiran, *Network calculus*. Springer Verlag, Lecture Notes in Computer Science, LNCS 2050, 2001.

[3] C.S. Chang, *Performance guarantees in communications networks*, Springer, 2000.

[4] R. Cruz, A calculus for network delay, part I : Network elements in isolation. *IEEE Transactions on Information Theory*, 37(1):114–121, 1991.

[5] R. Cruz, A calculus for network delay, part II : Network analysis. *IEEE Transactions on Information Theory*, 37(1):121–141, 1991.

[6] E. Knightly and N. Shroff, Admission control for statistical QoS: Theory and practice. *IEEE Network*, 13(2):20-29, March 1999.

[7] C. Li, A. Burchard and J. Liebeherr, Statistical Network Calculus for Multiplexed Arrivals, Technical Report, University of Virginia, Computer Science Department 2003.

[8] J. Liebeherr, S. Patek and A. Burchard, Statistical Per-Flow Service Bounds in a Network with Aggregate Provisioning. *Proceedings of the INFOCOM*, 2003.

[9] R. Nelson, *Probability, Stochastic Processes, and Queueing Theory*, Springer-Verlag, 1995.

[10] M. Reisslein, K.Ross and S. Rajagopal, A framework for guaranteeing statistical qos. *IEEE Transactions on Networking*, 10(01):27–42, 2002.

[11] M. Schwartz, *Broadband Integrated Networks*. Prentice-Hall, 1996.

[12] H. Zhang, Service disciplines for guaranteed performance service in packet switching networks. *Proceedings of the IEEE*, 83:1374–1399, 1995.

A New Model for Location-Dependent Semantic Cache Based on Pre-Defined Regions

Heloise Manica
Departamento de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)
Florianópolis - SC, Brasil
Heloise@inf.ufsc.br

and

Murilo S. de Camargo
Departamento de Informática – Universidade de Brasília (UNB)
Brasília - DF, Brasil
murilo@cic.unb.br

and

Ricardo R. Ciferri, Cristina D. A. Ciferri
Departamento de Informática – Universidade Estadual de Maringá (UEM)
Maringá - PR, Brasil
rrciferri@uol.com.br

Abstract

Mobile Computing is an emerging paradigm that provides to mobile clients the capability of accessing information anywhere and anytime. Data Management in this paradigm poses new challenging problems to the database community. New research problems include management of location dependent data. Location-Dependent Services (LDS) is an emergent application that allows new types of queries such as location-dependent queries and continuous queries. Data caching plays a key role in data management due to its ability to improve system performance and availability limitations. However, data cached in LDS can become obsolete when a mobile client moves from a location to a new one. The spatial property of location-dependent data opens up new challenges and opportunities for data caching research. The cache management requires more than the traditional solutions because mobility and location must be addressed. In this paper, we first provide a review of the existing approaches for data cache management in location dependent systems. Secondly, we propose a new model for location-dependent semantic cache. For this model, we present a new cache organization based on pre-defined regions, an improved cache replacement policy called ASCR and a strategy to build new semantic segments for LDS.
Keywords: Database**,** Cache Management, Mobile Computing, Semantic Cache.

1. INTRODUCTION

Advances in mobile computing and wireless networks provide flexibility in accessing and manipulating information. Database servers and mobile clients communicate via wireless networks without restricting users to specific locations. While this technology is revolutionary in its approach to managing data, many issues remain to be addressed. Traditional approaches may not perform well and more effective solutions by taking into account the impacts of mobile computing must be developed.

Data caching plays a key role in data management due to its ability to improve system performance and availability limitations. Undoubtedly, caching is advantageous to such environment since it helps to save power consumed with server communication. If queries results can be obtained locally, server communication is not required. Besides that,

wireless links are relatively unreliable and limited. This implies that the traffic should be reduced to improve system performance.

Another relevant issue is the client disconnections that can be voluntary or not. The cache technique helps to deal with disconnections since mobile users are still able to work using the cached data when the network is unreachable.

Mobility raises new requirements for data management. The transaction model, query processing mechanism, consistency control and recovery for the traditional distributed systems have to be adapted to be used in mobile environments [9].

Mobility also makes possible the development of new classes of applications such as Mobile Location-Dependent Information Services (LDIS). Nowadays, these applications are becoming popular in the mobile computing environment. Through these services mobile clients can access location sensitive data such as local traffic report, hotel and restaurant information, emergency services, etc. These access data, called Location Dependent Data (LDD), are related to their geographical position, and the queries that issue for them are named Location-Dependent Queries (LDQ). LDQ usually originates from moving objects, whose locations determine the results of these queries. "Find the closest restaurant within 10 miles" is an example of a LDQ. Answering this type of query requires the client's current position. The client's geographical position is provided by a Location Service or GPS-solution.

Similarly, the caching technique is also crucial for LDD applications. Moreover, data cached in LDIS can become obsolete when a mobile client moves from a location to a new one. Then, the cache management requires more than the traditional solutions because mobility and location must be addressed. There are important works in cache management for LDD ([2], [10], [11], [12]). However, how to choose an effective caching model still needs further study. The spatial property of location-dependent data opens up new challenges and opportunities for data caching research.

There are two issues involved in client cache management: cache invalidation that maintains data consistency between the client cache and the server; cache replacement policy that determines which data should be deleted from the cache when it does not have enough free space to accommodate a new item [12]. In recent years, cache consistency has been extensively studied; however most of the previous works have studied the cache invalidation problem incurred by data updates (temporal invalidation).

Since clients are not fixed equipments, the data stored in client's cache may be obsolete not only due to updates performed on data items but also when the client moves from one location to another. Spatial invalidation occurs when the data values stored in the client's cache become invalid because the client has moved to a new location area.

The maintenance of a valid cache when the client moves is called Location-Dependent Cache Invalidation and a data item can have different value depending on its location [12]. When the cache is full, a cache replacement policy has to be used to identify and delete the data that is most unlikely to be used again. In LDIS, traditional policies such as LRU and LFU are not suitable and must be adapted.

To better explore the spatial property, the semantic cache model has been studied for mobile computing environment ([1], [4], [7], [8]). This significant model and its advantages for mobile computing are discussed in section 4.

A comprehensive discussion and comparison of caching strategies for general mobile computing systems can be found in [5]. In a previous paper, we have presented a wide review in the cache management area. This paper extends our previous work to address caching issues for LDIS. The main contributions of our work are summarized as follows.

We first provide a review of the existing approaches for data cache management in location dependent systems. The review provides a basis for identifying strengths and weaknesses of individual methodologies, as well as general guidelines for future improvements and extensions.

Secondly, we propose a new model for location-dependent semantic cache. For this model, we present a new cache organization based on pre-defined regions, an improved cache replacement policy called ASCR and a strategy to build new semantic segments for LDIS.

The remaining of this paper is organized as follows. Sections 2 and 3 present location-dependent cache invalidation and replacement strategies proposed in the literature for the traditional page cache model. Another important solution, the semantic cache model is presented in section 4. In section 5 we propose a new semantic cache management based on geographic information that improves the query processing and the cache hit. Finally, section 5 discuss our proposed model and concludes this paper.

2. LOCATION-DEPENDENT CACHE INVALIDATION STRATEGIES

In a location-dependent information system, a data item can show different values for different geographical locations. That is, the answer to a query depends on the location where the query originates. This section and section 3 present location-dependent cache management strategies that are suitable for the physical cache model. In the physical data storage model, the MU cache contents are copies (tuple or page) of data items from the server. Page caching is a traditional approach and widely used in client-server systems [3].

Otherwise, another important model is the logical cache where arbitrary query answers are stored in the client's cache. Different from the physical model, the data is retrieved from the server using queries. This requires more processing capability at the server, but only the required data is transmitted over the wireless link. This type of cache model is described on section 4.

Following, we discuss location-dependent invalidation strategies. A common way to perform location-dependent cache invalidation is to attach validity information (named valid scope or valid area) to the data values in the cache. The valid scope of an item value is defined as the geographic area in which the item value is valid. There are different forms to relate the data value with its valid area. The form to represent the valid scope depends on the location model employed. In the symbolic model the valid scope is represented by a set of logical identifications (IDs), e.g. the ID of a cell in a cellular communication system. A geometric model represents a valid scope by geographical coordinates of the area, e.g. a polygonal.

Efficient cache invalidation methods are critical to the whole system performance. The attached information provides a way to check the validity of cached data with respect to a certain location without communicating with the server. Besides, the invalidation information can be used by cache replacement policies.

In [11] the validity information of a data item is the set of cells (symbolic model) within the item is valid. It is assumed that a client cache is logically organized and each cache entry contains a data item ID, attached validity information if any, and a pointer pointing to the real data.

The server delivers the valid scope along with a data item value to a MU. The client caches the data as well as its valid scope for later validity checking. For validity information organization [11] proposed three methods: Bit Vector (BV), Grouped Bit Vector with Compression (GBVC), and Implicit Scope Information (ISI).

The method BVC considers that each cell has identification (CID) and it uses a bit vector to record scope information. The bit vector length is equal to the number of cells in the system and all data in cache is associated to a bit vector to record the valid scope. This way, the value 1 (one) in the nth bit means that the data item value is valid in the nth cell while 0 (zero) means that it is invalid.

Whenever a data item value is required for location-dependent validation, the client listens to the broadcast for the current cell's ID, and uses it to examine the cached bit vector of the data item value. When the system is very large, the overhead is large.

To remedy the overhead in BVC, the GBVC only store information about cells that are adjacent or near MU current location. The model proposes the division of the wide geographical area of the system into groups and intra-groups. The cell ID consists of two parts: group ID and subgroups ID.

When a mobile client checks the validity of a data item value, it listens for the current cell's ID, i.e., (group-IDc, intra-group-IDc), and compares the group-IDc with the one associated with the cached data. If they are not the same, the data is invalid. Otherwise, the client checks the intra-group-IDcth bit in the bit vector to determine whether the cached data is valid.

The Implicit Scope Information model divides the database into multiple logic sections. Data items with the same valid area are placed at the same section. The data item in cache will have the format {Di, SDNi and SNi}, where Di is the data item value, SDNi is the section number, and SNi the data number inside the section (scope number).

In [12] is described tree schemes for representing valid scopes: Polygonal Endpoints (PE), Approximate Circle (AC) and Caching-Efficiency-Based (CEB).

The PE strategy records all endpoints of the polygon representing the valid scope of a cached data item. When the number of endpoints is large, this technique will consume a large portion of the wireless bandwidth and space for caching the valid scope in the client. The advantage is the complete knowledge of the valid scopes.

Another alternative is the utilization of an approximate circle (valid area circle) inserted inside the original polygon (v1' in figure 1). Thus, the valid area will be the approximate area defined by the center and radius of the circle. A problem occurs when the polygon shape is thin and long. In this case, the approximating error will be high and the cache can consider valid data as invalid if the query is outside the circle.

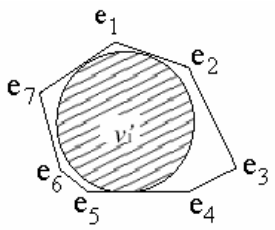


Figure 1: Possible Valid Areas. From [12]

Caching-Efficiency-Based is a generic method for balancing the overhead and the precision of the valid scopes to be attached. CEB generates a candidate valid scopes set, and then select the best one.

3. LOCATION-DEPENDENT CACHE REPLACEMENT STRATEGIES

Due to limitations of the client cache size, it is impossible to hold all the accessed data items in the cache. As a result, cache replacement algorithms are another important issue to be handled.

The MU location changes open up new research issues in replacement policies. In LDIS, the cache replacement policy must consider other factors besides the traditional factor "access probability". Policies such as LRU, LFU, and LRU-K are not suitable for this application.

When mobile client moves around, other factors must be considered such as valid scope, distance and direction. Valid scope represents the geometric area in which the data value is valid. A common way to perform location-dependent cache invalidation is to attach the valid scopes to the data values returned to the client [10]. The larger the valid scope area of the data, the higher the probability that the client requests this data.

In LDIS, the server answers queries according to the client's location, then the distance is an important factor to be considered. When the valid scope of a data value is far away from the current client's location, this data will have a lower chance to become useful. The computation of the distance between client's location and the data valid area can change according to the kind of application. In a rural zone for example, we can use the Euclidean distance $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. However, in an urban area this formula is not suitable because the MU can move through the streets with buildings or other obstacles.

Another used factor is the direction. It can be used first to eliminate from the cache the data that are in the opposite direction of the client's movement. To use this factor a mobility model is required to represent the locations and moving behaviors for mobile users.

Next we briefly present some cache replacement policies for location-dependent systems. These methods were proposed for physical cache model (page or tuple cache).

The Probability Area (PA) proposed in [12] defines the data items to be replaced according to a cost function defined as the product of the access probability of a data item and its attached valid scope. The cost function of a value j of item i is: $C_{i,j} = Pi \cdot A(v_{i,j})$, where Pi is the access probability of the item i and A(v'i,j) is the area of the attached valid scope for a value j of an item i. When the data replacement is performed, this policy selects the data with the least cost (s).

In the Probability Area Inverse Distance (PAID) policy [12], the cost function of a value j of an item i is given by C i,j = Pi . A(v'i,j) / D(v'i,j) where Pi and A(v'i,j) is defined in the same way as above, and D(v'i,j) is the distance between the current location and the valid area v'i,j. When data replacement is carried out, PAID ejects the data value (s) with the least cost (s).

To consider the movement direction, the authors in [12] take extensions of the model PAID: PAID-U (Probability Area Inverse Distance - Unidirectional) and PAID-D (Probability Area Inverse Distance - Directional). In PAID-D, the distance is calculated considering the client's current direction of movement. PAID-D keeps the data that are in the direction of the client's movement. On the other hand, in PAID-U the distance is computed regardless of the current direction of the client's movement.

In urban area, it is more difficult to find out buildings such as post office and hotel, is more general than issuing in rural area. Consequently, good services cannot be provided for users without considering the characteristics of urban area [13]. The Manhattan Distance policy was proposed by Jung et al. for location dependent queries in urban area. The distances in urban zones are given by |(x1-x2)|+|(y1-y2)|. The proposed algorithm computes the Cache Replacement Score (CRS) based on the MD computation. It chooses the victims according to CRS by the current location of the MU. Thus, the victims who are farthest from the MU will be replaced first.

4. SEMANTIC CACHING

As discussed before, the client cache technique plays an important role in mobile computing. Recent studies show that the semantic cache model (SC) is an attractive approach for mobile computing ([1], [4], [7], [8]).

The SC idea is to maintain in the client's cache both the semantic descriptions and associated answers for previous queries. The semantic information is very useful to organize and to manage the client's cache.

Affinity refers to the kind of relationship between the data items in the cache. This relationship can be temporal (temporal locality) or semantic (semantic locality). Temporal locality means that items referenced recently are likely to

be referenced again in a near future. Semantic locality means that if an item has been referenced, other items with the same semantic function (for instance, the nearest) are also likely to be referenced.

Semantic Cache is very useful in applications where data items do not change frequently and, hence, clients can cache data items and use them to serve queries locally. However, if the data are frequently updated, caching may not be helpful. In this case, broadcasting the data on the air may be a good solution.

The query processor uses the semantic descriptions to determine which data is available in the cache and which ones will have to be requested to the server. The semantic description is also utilized in the definition of the cache replacement policy, not requiring any additional attached information to each tuple as in traditional cache management systems.

In a basic approach, the semantic segment is represented by the set (SR, SA, SP, and SC), where SR and SA define the relation and the involved attributes. SP denotes the selection condition and SC represents the result (pointer for the result pages). Each semantic segment is associated with a pointer pointing to the first page in the cache memory.

The answer of a query Q can be totally contained in a segment S, partially contained or it cannot be answered for S. If Q can be partially answered by S, it is divided in two parts: probe query, which represents the portion of Q satisfied by S and remainder query, which represents the portion of Q not found in S. This procedure is called query trimming.

After the query splitting by the first segment, the next candidate segment will divide again the remaining query. This process continues until there are no more candidate segments or the reminder query becomes empty. At the end, if the reminder query is not empty it is sent to the database server to be processed. When the server returns the answer, the whole result is composed by all probe and reminder queries. A detailed study of semantic cache management in mobile computing can be found in [8]. The authors explore the semantic caching query processing strategies.

4.1 Semantic Caching for LDIS

Semantic caching is by nature an ideal cache scheme for location-dependent applications due to the following reasons [9]. First, semantic cache is built on the semantic locality among queries, which just fits the LDD applications. Secondly, continuous queries can be incrementally processed by semantic caching. Thirdly, since a semantic cache organizes data by semantic information, it makes cache management more flexible. A page or tuple cache cannot offer such functionality, since the cached data is not associated with any semantic meanings.

With semantic information, different strategies can be developed. If the user provides information about your schedule and route, it is possible to define the exact data items that the user will need later. However, when user-provided information is not unavailable, strategies need to be carefully designed to manage the location-dependent cache.

Most semantic cache invalidation and replacement strategies are built on temporal locality. LRU and MRU are examples of temporal locality invalidation strategies. For LDIS, the semantic locality is more appropriate.

[1] were the first to use semantic distance function for replacement policy. They utilize the Manhattan distance function to calculate the distance of each semantic region from the user's current location. Their proposed algorithm discard regions according to the values computed.

[6] extend the work of Dar et al. to investigate ways in which semantic caching can be used to manage location-dependent data. Their work includes a formal model to represent moving objects and propose strategies of applying semantic caching to location dependent applications. In this model, the semantic locality is highly related with the moving behavior of mobile users.

For cache replacement, Ren and Dunham [6] propose a semantic cache replacement policy called FAR (Furthest Away Replacement). FAR chooses replacement victims according to the user's status. The future location is calculated based on user's current location, moving speed and direction. Then, the segments are classified in two sets: the first one with segments that are in the direction of the movement, and the second with segments that are out of the direction. The victims are always selected from the out-direction set. When the out-direction set is empty, the most distant segments of the in-direction set are replaced. In the next section, we describe our proposed semantic cache model and cache management strategies.

5. LDD Semantic Cache Based on Pre-Defined Regions

The cache technique is only useful when the data cached is used to answer queries. If there is a cache hit, the client can serve the query locally; otherwise, it is necessary to send an uplink request to the server. The higher the cache hit ratio, the higher the local data availability, the less the uplink cost and battery consumption, and the less downlink cost

for query responses of fixed sizes [12]. So, query delay, bandwidth utilization and power consumption are related to the cache hit ratio.

There are techniques which can improve the cache hit ratio. In the prefetch technique, the clients prefetch data that may be used in the near future. Other important strategy to increase the cache hit ratio is the cache replacement policy. Mobility motivates the development of new cache replacement strategies built around location and movement. In a mobile environment, the location-dependent data cache must be replaced taking into account the impact of mobility, since the cache contents are required to move as the mobile unit moves to a new location.

A problem in LDIS is that mobile clients may have different movement patterns. As time passes by, the MU can change its moving behavior. Data distance may or may not affect cache performance, depending on the mobile client's movement and query patterns [12].

Since mobility pattern may change over time, the ideal cache replacement politics for LDD systems varies; it will depend on the client mobility model. Motivated by these needs we propose ASCR (Adaptive Semantic Cache Replacement Approach), a new method for semantic cache replacement.

This research assumes that a MU always moves in 2-dimensional space. To simplify the problem, we consider only location dependent queries on single relations and the supported operator in cache are selections. In the next subsections we describe our proposed model.

5.1 Model Organization

Location dependent queries are constructed with two kinds of predicates: traditional location unrelated predicates and location related ones. For example, the query "Give the hotels in Brasília with price < 100" is a location unrelated predicate, otherwise, the query "Give the hotels within 20 km from my actual position" is a location related. Our work considers the location related ones.

A location dependent query is defined as follows. Given a database D = {Ri}, a location dependent query Q is a tuple (QR, QP, QMBR, QC) where QR∈D represents the relation issued, QP a location predicate, QMBR the minimum bounding rectangle that represents the geographic area that is going to be queried and QC represent the query results.

In semantic cache model, the key structure is the semantic segments (or regions). We organize the answers received from the server in sets named segments. Basically, each segment has a set of tuples, a constraint formula that describes the tuples grouped together within the segment, and a replacement value. Each tuple in cache is associated with only one segment.

Our location dependent semantic model is basically composed by three main parts: the content, the index and the group structure. The content part consists of the queries results. The index part maintains the semantic as well as physical storage information for every cached segment. The index is represented by (S, SR, SP, SMBR, SG, SC, Sts) where S stands by the segment identifier, SR the database relation involved in the query, SP the select condition, SMBR the minimum bounding rectangle (IX=[X1, X2] and Iy[Y1, Y2]) that represents the geographic area the segment, SC the first page address of the segment content, Sts a timestamp, and SG the group. The group structure maintains the set of segments with related geographic position (table 2).

This model differs from other semantic cache schemes in the following aspects. First, we further propose to maintain spatial information, the rectangle that represents the geographic area of the data in the segment. This spatial information will be very useful in cache admission, replacement and query processing. Second, we propose an extra structure named "groups" that represents a geographic region previously defined in the system. In addition to the basic components listed above, other items can be kept for maintenance use.

The semantic cache index is more clearly illustrated through the following example 1. Let's consider tree database relations: Restaurant (Rno, Rname, Rtype, Rx, Ry), Hotel (Hno, Hname, Hvac, Hx, Hy) and Drugstore (Dno, Dname, Dschedule, Dx, Dy). Suppose that a MU issues queries on his path from different positions. The MU position is represented by M(x,y) and the following queries are issued on T1, T2, T3 respectively:

T1 - UM(40, 50): "Give me all hotels within 20 km"
T2 - UM(20, 40): "Give me all restaurants within 5 km and type is Chinese "
T3 - UM(30, 25): "Give me all Drugstores within 10 km"

Suppose that the client cache is empty and these queries results are cached. The index and group structures are formed as demonstrated in table1 and table2.

S	S_R	S_P	S_{MBR}	S_C	S_{ts}	S_G
S_1	Hotel	(UM_x -20 ≤ Hx ≤ UM_x + 20) ^ (UM_y - 20 ≤ Hy ≤ UM_y + 20)	[(20,80), (25,85)]	2	T1	1
S_2	Restaurant	(Rtipo = "chinese") ^ (UM_x -5 ≤ Rx ≤ UM_x + 5) ^ (UM_y - 5 ≤ Ry ≤ UM_y + 5)	[(15,25), (35,45)]	5	T2	1
S_3	Drugstore	(UM_x -10 ≤ Dx ≤ UM_x + 10) ^ (UM_y - 10 ≤ Dy ≤ UM_y + 10)	[(20,40), (15,35)]	8	T3	1

TABLE 1: Example of Semantic Cache Index

G_{ID}	G_{Seg}
A	S_1→ S_2→ S_3

TABLE 2: Example of Group Structure

5.2 Query Processing

The previous section defined our semantic cache model organization. This section describes important practical issues such as: query processing steps and the formation of semantic segments. As discussed before, the results of previous LDQ are cached in MU memory to be reused later. To reuse this information, not only the selecting conditions (predicate) but also the location must be satisfied. When a LDQ is issued, the first step is attaching to the predicate the user current location. After that, an LDD query can be processed normally using different strategies.

To efficiently verify the cache content and locate the candidate segments to answer queries, we propose first to verify which groups are candidates to answer the query. This is done through the analyses of intersected geographic area of the query (rectangle defined by MU current position and the dimension defined on the query predicate) with the geographic area of segments.

The same way, after selecting the candidate groups, the algorithm verifies witch semantic segments are candidate to answer the query. With the selected segments set, then is verified which of them have the same relation and predicate of the query. Finally, the query is executed in the selected segments. To locate candidate segments spatial index can be used.

This procedure is illustrated in example 2 (see Figure 2). Consider that different queries were issued and cached. Suppose that in T8 the client Mx,y issues the query Q= "Give me all hotels within 30 km". First the groups A and B will be selected. Secondly, the segments S3 and S4 are candidates. After verifying the relations between predicates, only S4 is candidate. Note that this case, only a small part of the total result is in cache. The intersected area contains the probe query results. The reminder query is submitted to the server for processing. The advantage of this method is that it avoids searches in segments that cannot answer the query. The spatial analysis eliminates all no-candidate segments.

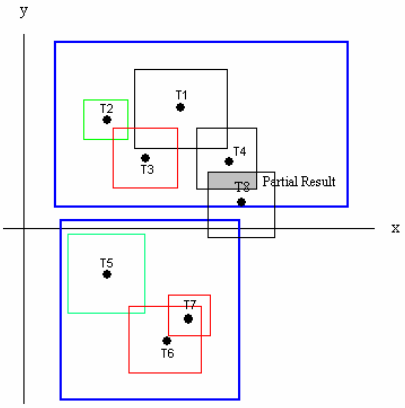


Figure 2: Example of Queries and Moving Path

5.2.1 Semantic Segments Formation

Another important issue related to query processing is the semantic segment formation and management. To avoid redundant data, we cannot keep S and Q in the cache. The way semantic regions are formed determines the granularity of the segments in cache and thus has a significant effect on the performance of the semantic cache. When a query intersects semantic segments in cache, different approaches can be used.

The no coalescing approach generates three disjoint parts: (Q ∩ S), (S ^ ¬Q) e (¬ S ^ Q). The disadvantage is that it may result in a large number of small segments. Besides, for our semantic cache model it would be too much difficult to represent the geographic area of a segment after it was trimmed for queries.

Complete coalescing approach always coalesces the three parts into one larger segment. This approach with small queries (relative to cache size) leads to good performance. However, when the answer takes a large portion of the cache, this strategy can result in semantic regions excessively large. This is not good for replacement due to the fact that a large segment can empty a significant portion of the cache resulting in poor cache utilization [1].

According to [8] the "partial coalescence in query" approach is the best one. The segment is decomposed in two parts: the overlapped part and the no overlapped part. Then, on must coalesce every part of Q together and cache the result of Q.

In this research, we use the "partial coalescence in query" approach and the idea proposed in [1]. An adaptive heuristic where regions formed at the same time were coalesced if either of them was smaller than 1% of the cache size. This way, the client can adjust the coalescing strategy dynamically.

The pre-defined groups represent urban areas. Our cache admission politics consider that when a MU is out of the pre-defined groups, this query should not be cached because those data are about a rural area and probably those data will not be used again.

5.3 Replacement Policy

In a mobile computing environment the users are equipped with mobile equipments which can have different mobility models. An optimal replacement policy discards the segments which will not be used in the future. For this, user-provided information is necessary, such as his movement, queries he is going to ask, future location and so on. Since this information is not always unavailable, the replacement policy must analyze the MU profile and define a better replacement.

One of the main motivations of our work is the desire to use geographic information to the cache management at client. This section describes our new replacement policy ASCR (Adaptive Semantic Cache Replacement Approach), a hybrid method for location dependent cache replacement based on temporal and semantic locality.

Before describing ASCR algorithm, let's consider the following examples 3 and 4 to illustrate different cache replacement scenarios. For both examples consider that the user begins issue location dependent queries in a yellow page database, and the client's semantic cache stores the results of the queries submitted. Consider also that the client's cache space is limited and when there is not sufficient space, a victim must be chosen to be replaced.

Example 3 - Low mobility: The user arrives in a city A, becomes a guest in a hotel and begins to issue queries for tourist information. During some time (maybe hours, days or weeks, etc.) while he visits the scenery he continues issuing LDD queries. When the cache replacement comes up, to eliminating data that are more distant from the user location may not be the best policy, since probably most of the cache content is about the local city. The LRU (least recent used) policy in this case is more suitable, simpler and avoids unnecessary and costly calculus to compute distance, future location and so on.

Example 4 - High mobility: The user is a traveler that visits many places. He begins his route in city A following to the cities B, C, D, an so on. He can issue some queries in all visited cities or not. Also it is possible that he comes back to any visited city. When the cache replacement comes up, for this case it is interesting first to eliminate data that are too distant from the user location. If the most distant data have already been eliminated and free space still is necessary, the next replacement can be done by the LRU policy.

For both exemplified cases, the user has no deterministic path and it is difficult to determine his real future location. ASCR defines the best policy according to the analyses of the previous client's movement. This can be done through the analyses of the group structure (table 2). Trough the MU current position, it's group can be easily identified (MUg). If there is more than one group in the structure, the older segments (ols) from the most distant group (mdsg) are replaced first. Otherwise, when there are segments from a single group, only the LRU is performed. Thus, distance computation is used only when there is more than one group.

To compute the distance between groups and the MU current position, different distance measure can be used. We assume the distance is defined as Euclidean calculus. This procedure is described in the algorithm ASCR (C, M), where C represent the client cache and M the mobile unit.

Algorithm: ASCR (C,M)
{ dsg ← NULL
For each *group* in C {
If UMg = *group*
then go to next *group*
else
dsg ← *dsg* + *group*}
if *dsg* = NULL
then LRU policy is performed over the semantic segments in cache
else GMD (*dsg*) }

GMD (*dsg*)
While (*dsg* != empty) {
Mdsg ← the most distant group from the *UMg*
For each segment S in *mdsg* {
ols ← the older segment from the group
discard *ols* from C
add data in free space
if (free space is enough) return (Success); }
if (free space still is enough) then LRU policy is performed over the single remaining group }

6. FINAL REMARKS AND CONCLUSIONS

In this paper we have explored cache management issues for location-dependent applications. We first presented different strategies proposed in the literature for the page and semantic cache model.

Compared to page caching, a semantic caching scheme is more efficient in location dependent applications since it uses semantic information to organize data. Table 4 [9] summarizes the advantages.

Issue	Semantic Cache (SC)	Page Cache	SC Advantage
Communication cost	low	high	Only required data are transferred.
Cache space	low	high	Only data satisfying previous queries are stored in SC
Parallelism in query processing	easy	difficult	Client and server work in parallel
Disconnection handling	efficient	inefficient	More autonomy of client since partial results can be obtained locally.
LDD data management	Efficient	inefficient	The management is done in semantic granularity.

TABLE 3: SC versus Page Caching

We have argued the need of new strategies that consider the spatial information on cache management issues. Based on this need, we present a new cache organization approach based on pre-defined regions, an improved cache replacement policy called ASCR and a strategy for build new semantic segments for LDIS. One of the main motivations of our work is the use of geographic information on client cache management to improve cache hit.

This model differs from other semantic cache schemes in the following aspects. First, we further propose to maintain spatial information, the rectangle that represents the geographic area of the data in the segment. This spatial information will be very useful in cache admission, replacement and query processing. Second, we propose an extra structure named "groups" that represents a geographic region previously defined in the system.

One of the main motivations of our work is the use of geographic information to the cache management at the client. We extend previous work in the following aspects. FAR policy is based on future locations of moving objects. Our model is based on previous locations. We don't use any mobility model, due to the fact that in a mobile environment

users have different path models. In order to verify the previous movement we use extra data structures. In FAR policy, each time the MU changes his direction, the sets in-direction and out-direction must be recalculated. To increase systems performance, our model avoids distance calculus. When cache replacement is necessary, our model utilizes its profile to decide the best replacement policy. Our politics considers that, besides the client is an equipment that can move, sometimes it can act as a stationary clients. In this case a policy based on temporal locality (e.g. LRU) is expected to perform well. Otherwise, if the client moves to different locations, a policy based on distance function is used.

FAR assumes that a MU moves at a speed that keeps constant during a period of time and may change from time to time. In some cases such as car moving in highways this model is suitable. Otherwise, in general an object can move anywhere with varied speed and direction. For instance, a car moving in an urban area, the velocity and mainly direction changes are very frequent. For future research, we will refine these ideas and use them in an evaluation study.

References

[1] Dar, S., Franklin, Michael J., Jónsson, Björn T. Srivastava, D., Tan, M. Semantic Data Caching and Replacement. Proceedings of the 22nd VLDB Conference Mumbai (Bombay), India, 1996.

[2] Dunham, Margaret H., Kumar, V. Location Dependent Data and its Management in Mobile Databases. Proceedings of the 9th International Workshop on Database and Expert Systems Applications, p.414, August 26-28, 1998.

[3] Franklin, M. J. Client Data Caching: A Foundation for High Performance Object Database Systems. Kluwer Academic Publishers, Norwell, Massachusetts, 1996.

[4] Lee, K.C.K., Leong, H.V. and Si, A. Semantic query caching in a mobile environment. In ACM Mobile Computing and Communications Review, Volume 3, number 2, pages 28-36, April 1999.

[5] Manica, H., Camargo, M. S. Caching Estrategies for Mobile Computing Systems. Proceedings of the 6th International Conference on Enterprise Information Systems. Universidade Portucalense, Porto – Portugal. April, 2004.

[6] Ren, Q.; Dunham M. H. Using Semantic Caching to Manage Location Dependent Data in Mobile Computing. Proceedings of Mobicom 2000: 210-221, May 2000.

[7] Ren, Q.; Dunham M. H.. Using Clustering for Effective Management of a Semantic Cache in Mobile Computing. MobiDE – Seattle – USA. ACM 1999.

[8] Ren, Q.; Dunham, M. H.; Kumar, V.. Semantic Caching and Query Processing. IEEE Trans. on Knowledge and Data Engineering, vol 15, n. 1, jan/feb 2003.

[9] Ren, Q. Semantic Caching in Mobile Computing. PhD Thesis presented to Souther Methodist University. May, 2000.

[10] Xu, J., Tang, X., Lee, D. L. and Hu, Q. Cache Coherency in Location-Dependent Information Services for Mobile Environment, Proc. the 1st Int. Conf. on Mobile Data Access (MDA'99), Hong Kong, Dec. 1999, Springer-Verlag LNCS, vol. 1748, pp. 182-193.

[11] Xu, J., Tang, X., Lee, D. L. Performance Analysis of Location-Dependent Cache Invalidation Schemes for Mobile Environments. TKDE 15 (2) 474-488, 2003.

[12] Zheng, B.; Xu, J.; Lee, D. L. Cache Invalidation and Replacement Strategies for Location-Dependent Data in Mobile Environments. IEEE Trans. on Computers, Special Issue on Database Management and Mobile Computing, 51(10): 1141-1153, October 2002.

[13] Jung, Y. Y.; Lee, J. and Kim, K., 2002. Broadcasting and Caching Policies for Location-Dependent Queries in Urban Areas. WMC'2002. Georgia, USA.

Modeling Transactions in UML Activity Diagrams via Nonsequential Automata

Júlio P. Machado
Pontifícia Universidade Católica do Rio Grande do Sul, Faculdade de Informática
Porto Alegre, RS, Brazil
juliopm@inf.pucrs.br

and

Paulo B. Menezes
Universidade Federal do Rio Grande do Sul, Instituto de Informática
Porto Alegre, RS, Brazil
blauth@inf.ufrgs.br

Abstract

When modeling concurrent or parallel systems, we must be aware that basic activities of each system may be constituted by smaller activities, i.e. transitions are conceptually refined into transactions. Nevertheless, the Unified Modeling Language (UML) seems to lack compositional constructs for defining atomic activities. We discuss nonsequential automata for the formal interpretation of the concept of composing transitions into transactions under UML activity diagrams. Transactions are formally defined through a special morphism between automata that maps transitions from the source automaton to transactions of the target (more concrete) automata. UML activity diagrams are then extended with a proper stereotype for defining transactions.

Keywords: Formal Specification, UML, Nonsequential Automata, Concurrent and Distributed Systems.

1 Introduction

The Unified Modeling Language (UML) [8, 3, 20, 10] is a language which may be used to describe both the structure and behavior of object-oriented systems using a combination of notations. UML offers a variety of graphical diagrams for specifying, visualizing and documenting object-oriented systems. These models can be classified as concerned with the static structure of systems and those concerned with the dynamic behavior. For the modeling of the dynamic behavior, a number of different models are offered: sequence, collaboration, statechart and activity diagrams.

UML has a precisely-defined syntax, however it still lacks a generally accepted formal semantics which precisely fixes the meaning of its diagrams. This is particularly noticeable for the notations which model the dynamic behavior of systems. Several approaches to translating UML diagrams into formal models have been based on Petri nets [17]. For example, [9] describes a formal translation of activity and collaboration diagrams into place/transition Petri nets and [7] compares different proposals for the semantics based on Petri nets. Also workflow models have been given semantics based on place/transition nets, e.g. [6] defines a formal semantics for UML activity diagrams that is suitable for workflow modeling. Widely application of Petri nets in theoretical and applied research to specify and visualize the behavior of systems are due to their intuitive graphical representation and several supporting tools.

Activity diagrams are one of the means for describing behavior of systems within the UML language and has been chosen here for its capability in describing flows of activities of a desired system. Broadly speaking, states of an activity diagram mostly represent the execution of sequential and concurrent steps in a computational process or a workflow step. Although activity diagrams have been used for modeling workflow processes [6, 7, 8, 3], such view is not going to be explored in this paper. Here we concentrate on describing groups of sequential or concurrent activities that are responsible for performing a computation, and we address the issue of modeling transactions, a feature not present in activity diagrams, by using a Petri net related model, named Nonsequential Automata ([11], [13], [15], [14]).

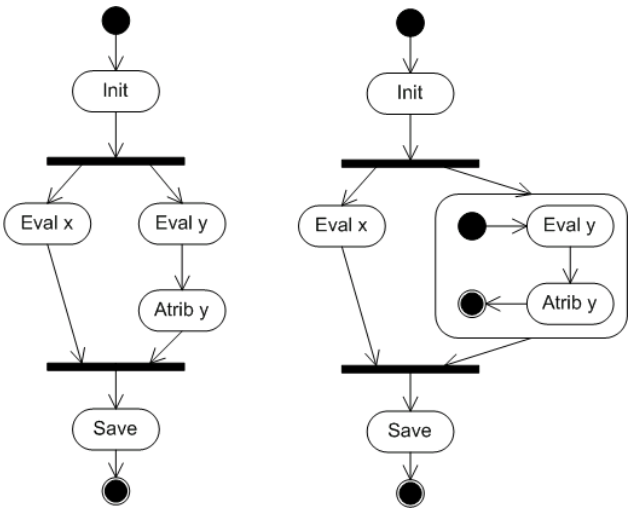


Figure 1: UML activity diagram without (left) and with (right) composite state

We remark that in our setting the term "transaction" denotes a certain activity of the system that might be composed by many, possibly concurrent, sub-activities. Moreover, we require this composition of activities to be considered atomic.

The importance of discussing composition mechanisms is based on the fact that, as pointed in [1], in order to master the complexity of systems, models demand a form of structural decomposition. When building large systems, one needs constructs to describe systems as compositions of other smaller systems. In this sense, when modeling a computational process, we need means of composing sub-activities both in a non atomic or atomic way.

All foundational work presented in this paper is developed within category theory [2], as it provides powerful techniques to unify different models through adjunctions expressing relations between their semantics [21],[2]. As the reader will see, adjunctions consist of ways of translating from one model to another.

The rest of the paper is organized as follows. Section 2 briefly recalls some basic definitions of UML activity diagrams and presents the need for the definition of atomic composition of activities (i.e. the definition of transactions). Section 3 presents the compositional semantic domain of nonsequential automata, which is going to be used as the semantics for the composition of activities in UML. Section 4 introduces translation schemes for building nonsequential automata from activity diagrams. Finally, Section 5 discusses the results and outlines possible directions for future investigations.

2 UML Activity Diagrams

Activity diagrams are one of the means for describing behavior of systems within UML. It defines an extended view of state machines focused on the flow of control from activity to activity, instead of statechart diagrams which is focused on potential states of objects and transitions among those states. Figure 1 depicts a simple example of an activity diagram for an operation.

For the most part, activity diagrams involve modeling sequential and concurrent steps in a computational process. In UML, those steps may be an action or a subactivity state. Action states represent an atomic computation without substructure and that cannot be interrupted. Subactivity states, on the other hand, represent non-atomic execution and can be decomposed and interrupted by external events (represented by transitions out of the state) - they can be think as a composite whose flow of control is made up of other activities and actions. A subactivity state is semantically equivalent to expanding its activity graph in place until there are only actions [10]. Nonetheless, they are important because they help breaking complex computations into parts. There is no difference in notation for action and subactivity states, besides the nested activity graph. Other important states are initial and final states.

Transitions between states are by means of arcs connecting the source and destination states. Transitions may be adorned with an event trigger, a guard condition and an action. The event trigger is the event whose reception by the source state makes the transition enabled. A transition without an explicit event is called a completion transition, indicating that it is enabled once the activity/action in the source state has been completed. A guard condition is a boolean expression that is evaluated when a transition is triggered by an event, causing the transition to fire if its value is true. Optionally, an action may be executed in response to the transition. To aid the visualization, branches are introduced to specify alternate paths based on boolean

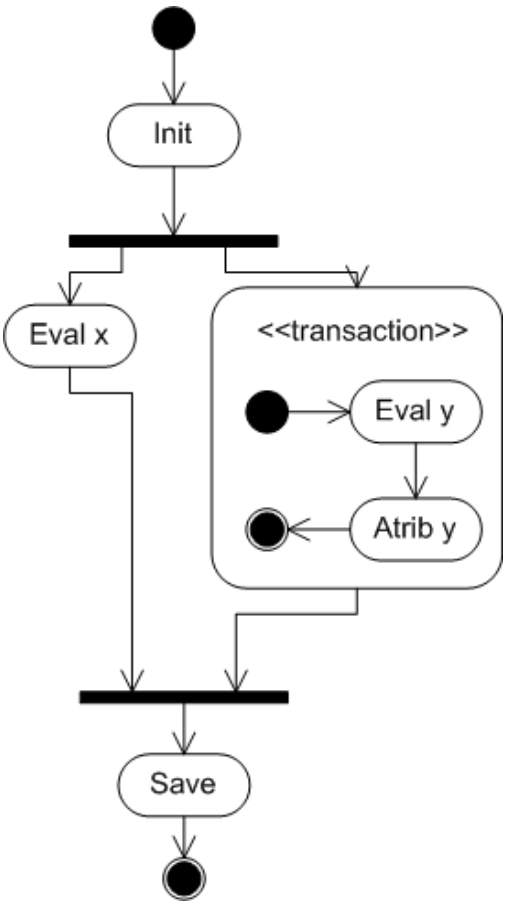


Figure 2: UML activity diagram with transaction composite state

expressions. A transition leaving an action state can be labeled with guard conditions and action expressions, but not with events. Normally, as stated in [10], an activity diagram will not present transitions with events, as activity diagrams assume that computations proceed without external event-based interruptions [1]. In order to capture complex transitions, horizontal bars represent the fork and join of concurrent steps of computation. Bellow the fork, the activities associated with each path continue in parallel. A join represents the synchronization of two or more concurrent flows of control, meaning that each waits until all incoming paths have reached the join.

In UML, activities are defined as non-atomic computations thus capable of being composed by subactivities. Borrowing the concept of composite state from general state machines, activity diagrams are provided a way of composing several actions/activities into new activities. A composite state may contain either sequential or concurrent substates. Nevertheless, the composite is in essence non-atomic, allowing transitions out of the composite with the effect of interrupting its entire computation process. In other words, activity diagrams lack a composite state for atomic composition of activities.

To overcome the lack of an atomic subactivity state, we introduce a new notation for a composite state based on the idea of atomic transaction. The notation is based on extension mechanisms of UML [3, 10] and introduce new constraints over the basic composite state, so the new "transaction state" is not allowed to be interrupted by explicit external events. The new composite state is decorated with the stereotype $<< transaction >>$ as depicted in figure 2. The semantics for the whole activity diagram and specially the transaction composite is presented in the next sections.

3 Nonsequential Automata

When employing Petri net based models to represent systems, one must be aware that basic activities of each component may involve smaller internal activities. In other words, transitions are conceptually refined into transactions.

A convenient top-down approach to represent transactions is to start with an abstract model and then

[1] In this case, ordinary state machines and collaboration diagrams are preferable.

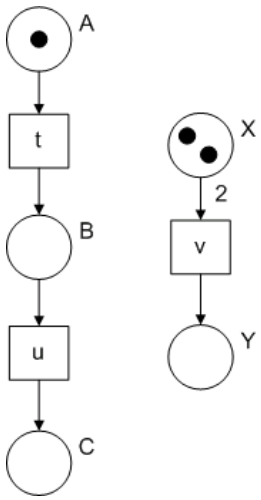


Figure 3: Marked place/transition Petri net

refine each transition, that might represent a complex activity, into a refined net that offers a more precise representation of the activity. Several approaches have appeared in the literature ([5], [12], [13]) that present different techniques for refinement. The approaches are related to the notion of general net morphism proposed by Petri, in which a refined net is collapsed into a transition of a more abstract counterpart. Here, we concentrate on the model named nonsequential automata.

Nonsequential Automata ([11], [13], [15], [14]) constitute a non interleaving semantic domain, with its foundations on category theory, for reactive, communicating and concurrent systems. Nonsequential Automata follows the so-called "Petri nets are monoids" approach [16] and is similar to Petri nets, but it is a more concrete model - it can be seen as computations from a given place/transition net.

Petri net, in this paper, means the general case of place/transition net with weighted arcs and without capacity restriction for places. So, we recall the definition of place/transition nets and then Petri nets as graphs in order to clarify the notation used for defining nonsequential automata. Also, for the sake of simplicity, we drop the notion of initial marking and initial/final states. The more interested reader is referred to [17] and [18] for more details.

Definition 1 (Place/Transition Net) *A place/transition net is a triple $N = \langle S, T, F\rangle$ where S is a set of places, T is a set of transitions, with S and T disjoints, and $F : (S \times T) \cup (T \times S) \to \mathbb{N}$ is the flow relation.*

The flow relation specifies how many tokens are consumed or produced in each place when a transition fires. A marking $m : S \to \mathbb{N}$, which is a finite multiset of places representing the number of tokens in each place, will be written $m = \{n_1A_1, \ldots, n_kA_k\}$ with n_i the number of tokens in place A_i (i ranging over $1, \ldots, k$) or as $n_1A_1 \oplus \ldots \oplus n_kA_k$ where $\oplus$ is the monoidal operation. For example, figure 3 represents the Petri net $N = \langle\{A, B, C, X, Y\}, \{t, u, v\}, \{(A, t), (t, B), (B, u), (u, C), (X, v), (v, Y)\}\rangle$ with initial marking $m = \{A, 2X\}$.

In order to define Petri net as a graph, which is more suitable to our discussion, we follow the same approach in [16], where nodes are elements of a commutative monoid. In this case, nodes and arcs stand for states and transitions of a net, where for each transition t, n_i tokens consumed in places A_i and n_j tokens produced in places B_j will be written the arc $t : \bigoplus n_iA_i \to \bigoplus n_jB_j$, for i and j ranging over $1, \ldots, k$. The transitions on the Petri net of figure 3 are represented by arcs $t : A \to B$, $u : B \to C$ and $v : 2X \to Y$. Therefore, a Petri net is basically a graph with a monoidal structure on nodes.

Before going any further, let us clarify all the semantic models in this paper are introduced as a class of objects equipped with a notion of behavior-preserving morphism, making each model into a category.

Definition 2 (Petri Category) *A (place/transition) Petri net is a graph $N = \langle S^{\oplus}, T, \delta_0, \delta_1\rangle$ where the set of nodes $S^{\oplus}$ is the free commutative monoid generated by the set of places and $\delta_0, \delta_1 : T \to S^{\oplus}$ are total functions called source and target of arcs in T. A Petri net morphism $h : N \to N'$ is a graph morphism $h = \langle h_S : S^{\oplus} \to S'^{\oplus}, h_T : T \to T'\rangle$ i.e. $h_S \circ \delta_i = \delta'_i \circ h_T$ for $i \in \{0, 1\}$ where h_S is required to be a monoid homomorphism. Petri nets as graphs and their morphisms define the category* ***Petri****.*

This view of Petri nets as graphs was based on the idea of nodes as elements of a commutative monoid over the set of places. The figure 4 depicts the behavior of the net in figure 3 when starting with a specific marking. What then if we change the initial marking? We have to compute all reachable markings again.

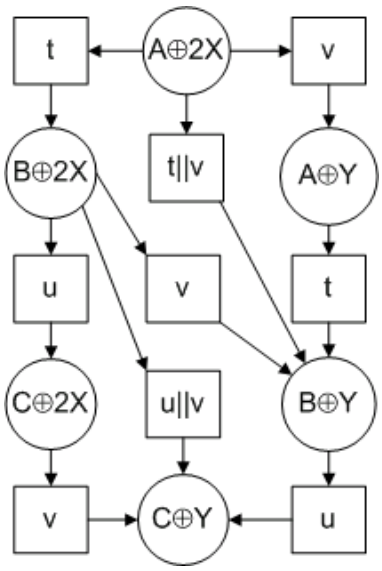


Figure 4: Behavior of a Petri net

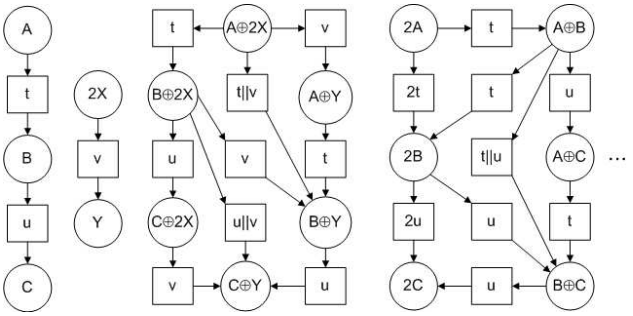


Figure 5: A nonsequential automaton corresponding to a Petri net

But what if we could get a more concrete model with all possible markings and capable of making explicit all implicit concurrencies in the net? This is the key for the nonsequential automata.

Nonsequential Automata constitute a categorical semantic domain around the concepts of state and transition. It consists of a reflexive graph with monoidal structure on both states and transitions, initial and final states and labeling on transitions. The interpretation of a structured state is the same as in Petri nets: it is viewed as a "bag" of local states representing a notion of tokens to be consumed or produced. A structured transition is a way to specify that the component transitions are independent i.e., structured transitions specify which component transitions are concurrent with another (see, for example, transitions $t||v$ and $u||v$ in figure 5).

In the next definitions **CMon** denotes the category of commutative monoids and $k \in \{0, 1\}$ (for simplicity, we omit that $k \in \{0, 1\}$).

Definition 3 (Nonsequential Automaton) *A nonsequential automaton* $NA = \langle \mathrm{V}, \mathrm{T}, \delta_0, \delta_1, \iota, \mathrm{L}, lab \rangle$ *is such that* $\mathrm{V} = \langle V, \oplus, 0 \rangle$, $\mathrm{T} = \langle T, ||, \tau \rangle$, $\mathrm{L} = \langle L, ||, \tau \rangle$ *are* ***CMon****-objects of states, transitions and labels respectively,* $\delta_0, \delta_1 : \mathrm{T} \to \mathrm{V}$ *are* ***CMon****-morphisms called source and target respectively,* $\iota : \mathrm{V} \to \mathrm{T}$ *is a* ***CMon****-morphism for mapping identities, and* $lab : \mathrm{T} \to \mathrm{L}$ *is a* ***CMon****-morphism for labeling transitions such that* $lab(t) = \tau$ *whenever there is* $v \in V$ *where* $\iota(v) = t$*. Therefore, a nonsequential automaton can be seen as* $NA = \langle \mathrm{G}, \mathrm{L}, lab \rangle$ *where* $\mathrm{G} = \langle \mathrm{V}, \mathrm{T}, \delta_0, \delta_1, \iota \rangle$ *is a reflexive graph internal to* ***CMon*** *representing the automaton shape,* L *is a commutative monoid representing the labels of the transitions and lab is the labeling morphism associating a label to each transition.*

In a nonsequential automaton, a transition labeled by τ represents a hidden transition. As the automaton shape is represented by a reflexive graph, each state has an associated identity transition which is interpreted as a "no operation" or "idle". Note that, by definition, all identity transitions are hidden (i.e. labeled by τ).

The nonsequential automaton in figure 5 is represented by $\langle \{A, B, C, X, Y\}^{\oplus}, \{t, u, v\}^{||}, \delta_0, \delta_1, \iota, \{t, u, v\}^{||}, lab \rangle$ with δ_0, δ_1, ι determined by transitions $t : A \to B$, $u : B \to C$, $v : 2X \to Y$, and labeling $t \mapsto t$, $u \mapsto u$,

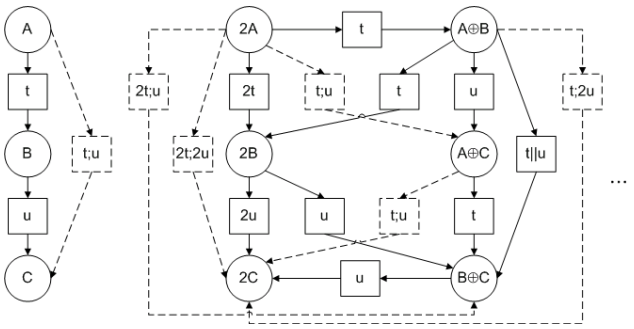


Figure 6: Nonsequential automaton and its corresponding computational closure

$v \mapsto v$. In the graphical representation of the automaton identity arcs are omitted and, for a given node A and arcs $t : X \to Y$ and $\iota_A : A \to A$, the structured arc $t||\iota_A : X \oplus A \to Y \oplus A$ will be simply noted by $t : X \oplus A \to Y \oplus A$. In this sense, in figure 5, the identity arcs $\iota_A, \iota_B, \iota_C, \iota_{2X}, \iota_Y, \iota_{A\oplus B}, \ldots$ were omitted, and transitions like $t||\iota_Y : A \oplus Y \to B \oplus Y$ where written $t : A \oplus Y \to B \oplus Y$.

Two important points must be discussed before going any further. First, the nonsequential automaton in figure 5 was not completely drawn as it has infinite distinguished nodes, for they are elements of a freely generated monoid chosen to represent its states. This choice was not obligatory as the definition of nonsequential automaton mentions any commutative monoid. This is different from other Petri net related models in which the monoid must be freely generated. Second, structured transitions, like $t||v$, explicitly determines the "independence square", i.e. transitions t and v are independent, and should not be confused with synchronized transitions presented by Meseguer and Montanari [16].

Definition 4 (NAut Category) *A nonsequential automaton morphism $h : NA \to NA'$ with $NA = \langle \mathrm{V}, \mathrm{T}, \delta_0, \delta_1, \iota, \mathrm{L}, lab\rangle$ and $NA' = \langle \mathrm{V}', \mathrm{T}', \delta_0', \delta_1', \iota', \mathrm{L}', lab'\rangle$ is a triple $h = \langle h_V, h_T, h_L\rangle$ with $h_V : \mathrm{V} \to \mathrm{V}'$, $h_T : \mathrm{T} \to \mathrm{T}'$, $h_L : \mathrm{L} \to \mathrm{L}'$ **CMon**-morphisms, such that $h_V \circ \delta_k = \delta_k' \circ h_T$, $h_T \circ \iota = \iota' \circ h_V$ and $h_L \circ lab = lab' \circ h_T$. Nonsequential automata and their morphisms constitute the category **NAut**.*

Given an appropriate notion of morphisms between two nonsequential automata defined above, we are able to define atomic composition of transitions through the concept of refinement. A refinement maps transitions into transactions reflecting an implementation of an automaton on top of another, based on constructions presented in [15]. It is defined as a special morphism of automata where the target one (more concrete) is enriched with its computational closure (all the conceivable sequential and nonsequential computations that can be split into permutations of original transitions). The construction of the computational closure was inspired by [16]. The idea for building the computational closure here is analogous to the example previously discussed for state machines, except for the fact it also includes nonsequential computations besides sequential ones. Considering the nonsequential automaton in figure 5, its computational closure is partially depicted in figure 6 (added transitions were drawn with a dotted pattern). Please note a composition operator ";" appeared in the structured transitions.

The computational closure of a nonsequential automaton is formally defined as the composition of two adjoint functors between the **NAut** category and the category **CNAut** (defined in the next paragraphs) of nonsequential automata enriched with it computations: the first one basically enriches an automaton with a composition operation on transitions, and the second functor forgets about the composition operation. Also, some equations about concurrency are introduced. This composition leads to an endofunctor called transitive closure functor tc. Then, the refinement morphism φ from NA into (the computations of) NA' can be defined as $\varphi : NA \to \mathsf{tc}NA'$.

In the text that follows, the categories are built using the approach known as internalization [2], leading to the notion of structured (internal) graphs, where nodes and arcs may be objects of different categories. The category of categories internal to **CMon** (remember **CMon** is the category of commutative monoids) is denoted by **Cat(CMon)** and **RGr(CMon)** is the category of reflexive graphs internal to **CMon**.

Definition 5 (CNAut Category) *Consider the category **Cat(CMon)**. The category **CNAut** is the comma category $id_{Cat(CMon)} \downarrow id_{Cat(CMon)}$ where $id_{Cat(CMon)}$ is the identity functor in **Cat(CMon)**. Therefore, a **CNAut**-object is a triple $CNA = \langle \mathsf{G}, \mathsf{L}, lab\rangle$ where G, L are **Cat(CMon)**-objects and lab is a **Cat(CMon)**-morphism.*

Notice that in order to build the computations, we have enriched **NAut** by the substitution of its shape from a reflexive internal graph $\mathrm{G} = \langle \mathrm{V}, \mathrm{T}, \delta_0, \delta_1, \iota\rangle$ to a **Cat(CMon)**-object $\mathsf{G} = \langle \mathrm{V}, \mathrm{T}, \delta_0, \delta_1, \iota, ;\rangle$ with a

composition operation, and similarly with its labels. The composition operation ";" was responsible for the newly added transitions in figure 6. This construction is formally defined by the following functor:

Definition 6 (Functor nc) *Let $NA = \langle \mathrm{G}, \mathrm{L}, lab\rangle$ be a **NAut**-object and $h : NA \to NA'$ be a **NAut**-morphism. The functor $\mathsf{nc} : \textbf{NAut} \to \textbf{CNAut}$ is such that:*

- ***RGr(CMon)**-object $\mathrm{G} = \langle \mathrm{V}, \mathrm{T}, \delta_0, \delta_1, \iota\rangle$ is taken into the **Cat(CMon)**-object $\mathsf{G}' = \langle \mathrm{V}, \mathrm{T}', \delta_0', \delta_1', \iota', ; \rangle$ with ι' induced by ι and T', δ_0', δ_1', $\_;\_ : \mathrm{T}' \times \mathrm{T}' \to \mathrm{T}'$ inductively defined as follows*

$$\frac{t{:}a\to b\in \mathrm{T}}{t{:}a\to b\in \mathrm{T}'} \qquad \frac{t{:}a\to b\in \mathrm{T}' \;\; u{:}b\to c\in \mathrm{T}'}{t;u{:}a\to c\in \mathrm{T}'} \qquad \frac{t{:}a\to b\in \mathrm{T}' \;\; u{:}c\to d\in \mathrm{T}'}{t||u{:}a\oplus c\to b\oplus d\in \mathrm{T}'}$$

  subject to the following equational rules

$$\frac{t\in \mathrm{T}'}{\tau;t=t \;\; t;\tau=t} \qquad \frac{t{:}a\to b\in \mathrm{T}'}{\iota_a;t=t \;\; t;\iota_b=t} \qquad \frac{t{:}a\to b\in \mathrm{T}' \;\; u{:}b\to c\in \mathrm{T}' \;\; v{:}c\to d\in \mathrm{T}'}{t;(u;v)=(t;u);v}$$

$$\frac{t\in \mathrm{T}' \;\; u\in \mathrm{T}'}{t||u=u||t} \qquad \frac{t\in \mathrm{T}'}{t||\tau=t} \qquad \frac{\iota_a\in \mathrm{T}' \;\; \iota_b\in \mathrm{T}'}{\iota_a||\iota_b=\iota_{a\oplus b}} \qquad \frac{t\in \mathrm{T}' \;\; u\in \mathrm{T}' \;\; v\in \mathrm{T}'}{t||(u||v)=(t||u)||v}$$

- ***CMon**-object L is taken into the **Cat(CMon)**-object $\mathsf{L}' = \langle 1, \mathrm{L}', !, !, !_\iota, ; \rangle$ with L' inductively defined as above, and $!$ and $!_\iota$ meaning the unique obvious mappings.*

- *The **NAut**-object $NA = \langle \mathrm{G}, \mathrm{L}, lab\rangle$ is taken into the **CNaut**-object $CNA = \langle \mathsf{G}', \mathsf{L}', lab'\rangle$ where lab' is the morphism induced by lab.*

- *The **NAut**-morphism $h = \langle h_V, h_T, h_L\rangle$ is taken into the **Cat(CMon)**-morphism $h = \langle h_G, h_L\rangle : CNA \to CNA'$ where $h_G = \langle h_V, h_{T'}\rangle$, $h_L = \langle !, h_{L'}\rangle$ and $h_{T'}$, $h_{L'}$ are the monoid morphisms generated by the monoid morphisms h_T and h_L, respectively.*

Definition 7 (Functor cn) *Let $CNA = \langle \mathsf{G}, \mathsf{L}, lab\rangle$ be a **CNAut**-object and $h : CNA \to CNA'$ be a **CNAut**-morphism. The functor $\mathsf{cn} : \textbf{CNAut} \to \textbf{NAut}$ is such that:*

- ***Cat(CMon)**-object $\mathsf{G} = \langle \mathrm{V}, \mathrm{T}, \delta_0, \delta_1, \iota, ; \rangle$ is taken into the **RGr(CMon)**-object $\mathrm{G}' = \langle \mathrm{V}, \mathrm{T}', \delta_0', \delta_1', \iota'\rangle$, where T' is T subject to the equational rule*

$$\frac{t{:}a\to b\in \mathrm{T}' \;\; u{:}b\to c\in \mathrm{T}' \;\; t'{:}a'\to b'\in \mathrm{T}' \;\; u'{:}b'\to c'\in \mathrm{T}'}{(t;u)||(t';u')=(t||t');(u||u')}$$

  and δ_0', δ_1', ι' are induced by δ_0, δ_1, ι, restricted to T'.

- *The **Cat(CMon)**-object $\mathsf{L} = \langle \mathrm{V}, \mathrm{L}, \delta_0, \delta_1, \iota, ; \rangle$ is taken into the **CMon**-object L', where L' is L subject to the analogous equational rule.*

- *The **CNAut**-object $CNA = \langle \mathsf{G}, \mathsf{L}, lab\rangle$ is taken into the **NAut**-object $NA = \langle \mathrm{G}', \mathrm{L}', lab'\rangle$ with lab' the **RGr(CMon)**-morphism canonically induced by the **Cat(CMon)**-morphism lab.*

- *The **CNAut**-morphism $h = \langle h_G, h_L\rangle$ with $h_G = \langle h_{G_V}, h_{G_T}\rangle$, $h_L = \langle h_{L_V}, h_{L_T}\rangle$ is taken into the **NAut**-morphism $h = \langle h_{G_V}, h_{G_{T'}}, h_{L_{T'}}\rangle : NA \to NA'$ where $h_{G_{T'}}$ and $h_{L_{T'}}$ are the monoid morphisms induced by h_{G_T} and h_{L_T} respectively.*

Definition 8 (Transitive Closure Functor) *The transitive closure functor is $\mathsf{tc} = \mathsf{cn} \circ \mathsf{nc} : NAut \to NAut$.*

The functor cn introduces an important requirement about concurrency in the computational closure: $(t;u)||(t';u') = (t||t');(u||u')$. That is, the computation determined by two independent composed transitions $t;u$ and $t';u'$ is equivalent to the computation whose steps are the independent transitions $t||t'$ and $u||u'$. As illustration, consider transitions $t : A \to B$ and $u : B \to C$ from our working example (figure 6). For $t||u : A \oplus B \to B \oplus C$ we have

$$t||u = (\iota_A;t)||(u;\iota_C) = (\iota_A||u);(t||\iota_C) = u;t$$

$$u||t = (\iota_B;u)||(t;\iota_B) = (\iota_B||t);(u||\iota_B) = t;u$$

Thus, the concurrent execution of two independent transitions is equivalent to its sequential execution in any order. Also, the inference rules of the functor nc inductively defined the new set of composed transitions, adding, for example, the transition $t;2u : A \oplus B \to 2C$ by sequentially composing $t : A \oplus B \to 2B$ and

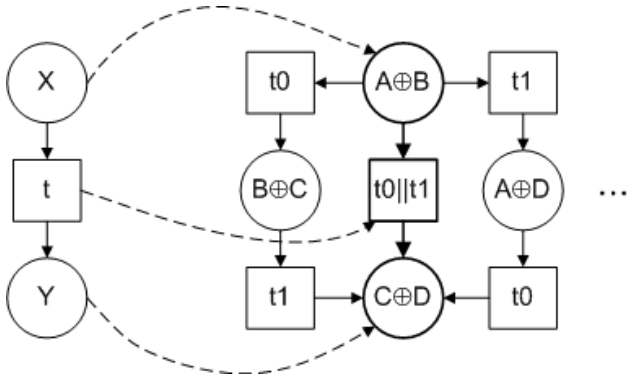


Figure 7: Refinement morphism for transaction $t : X \to Y$

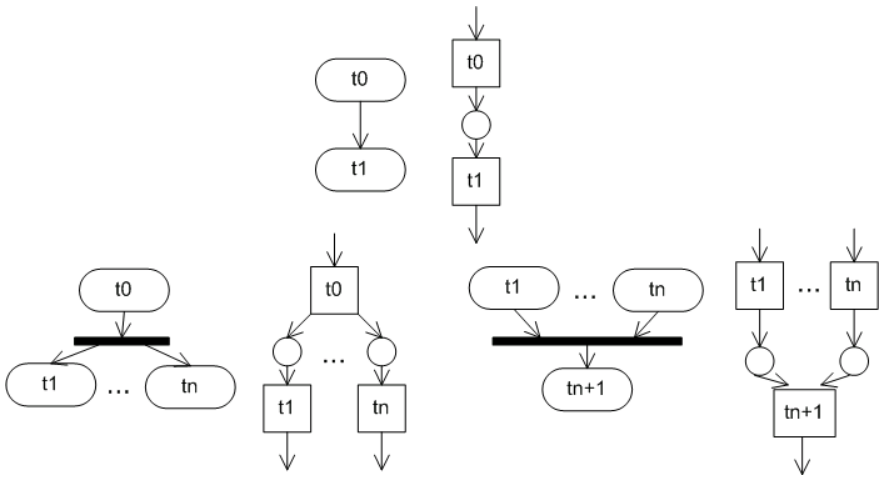


Figure 8: Basic translation schemes from activity diagram to nonsequential automaton

$2u : 2B \to 2C$. But how about transition $u;t;u : A \oplus B \to 2C$? It was not drawn in the figure because several transitions actually represent equivalence classes induced by the set of equational rules. In the case of $u;t;u$ it is in the same class of $t;2u$:

$$t;2u = t;(u||u) = (t||\iota_B);(u||u) = (t;u)||(\iota_B;u) = (t;u)||u = u||(t;u) =$$

$$(u;\iota_C)||(\iota_A;(t;u)) = (u||\iota_A);(\iota_C||(t;u)) = u;(\iota_C||(t;u)) = u;((t;u)||\iota_C) = u;t;u$$

To illustrate the refinement morphism, given two nonsequential automata NA and NA' with free monoids on states and labeled transitions respectively induced by transitions $t : X \to Y$, and $t_0 : A \to C$, $t_1 : B \to D$, suppose we want to build a transaction containing both t_0 and t_1. First we apply the transitive closure functor tc, enriching NA' with all sequential and nonsequential computations. For the last step we need to build the refinement morphism by mapping the corresponding states and transitions. The refinement $\varphi : NA \to \mathsf{tc}NA'$ is given by $X \mapsto A \oplus B$, $Y \mapsto C \oplus D$, $t \mapsto t_0||t_1$ (see figure 7). Notice that due to the equations, we actually get a class of transitions containing $t_0||t_1$, $t_0;t_1$ and $t_1;t_0$, represented as $t_0||t_1$ in the figure.

4 Mapping Activity Diagrams to Nonsequential Automata

As we can see from previous sections, an activity diagram is already close to a Petri net. So, the basic translation schemes from activity diagrams into nonsequential automata are targeted into constructing local transitions for a nonsequential automaton.

Figure 8 shows this principle for action states and complex transitions. The labeling for transitions corresponds to labels from the diagram states. Initial and final states represent the corresponding initial and final places/states of the automaton [2].

[2]The explicit representation of initial and final states for nonsequential automata was not presented in order to simplify the discussion of the semantic model.

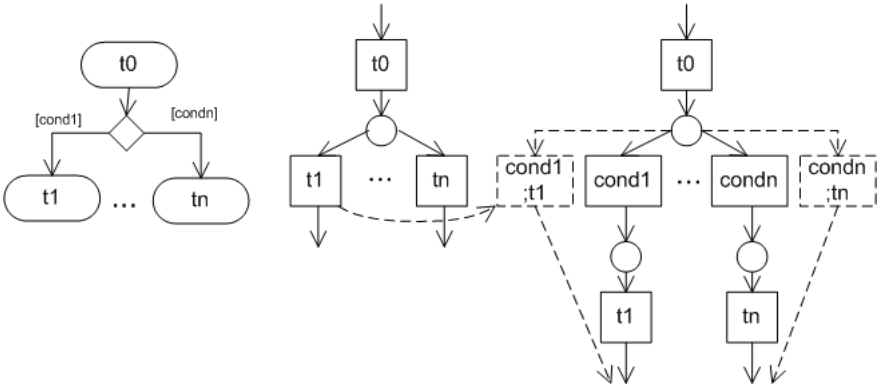


Figure 9: Translation scheme for guarded transitions

The translation of guarded transitions is based on the refinement morphism from nonsequential automata. It is a two step translation: first, conditions are translated to transitions corresponding to a nondeterministic choice following figure 9, then, a refinement associates each flat transition t_i to a sequential composition $cond_i; t_i$.

The central core of the composite transaction state also makes use of nonsequential automata refinement. The source automaton corresponds to the basic translation using the previous mappings, where the composite state is viewed as only one state. The target automaton corresponds to the translation taking into account the subactivity states of the composite. The refinement then maps the more abstract transition into the concrete implementation of the transaction obtained via the computational closure of the target automaton. Figure 10 partially depicts the target nonsequential automaton for our initial example of activity diagrams (figure 1). Notice it explicits all possible computational paths, including the transition labeled $Evalx||(Evaly; Atriby)$ as the desired behavior for the fork/join construction in the activity diagram. For the refinement, the transaction state is represented by the atomic composition $Evaly; Atriby$.

Due to space limitations and to simplify the presentation, in this paper we have not dealt with event generation and handling. Generally speaking, for Petri net related models, events may be modeled as tokens or transitions with different consequences on the resulting behavior (see [7] for a discussion on both alternatives). Notice when modeling reactive systems, events should not be abstracted away and thus activity diagrams alone are not sufficient in the development process.

5 Conclusion

Models capture the important aspects of the system being modeled from a certain point of view and simplify or omit the rest. Which aspects are essential is a matter of judgment that depends on the purpose of each model. We believe transactions are an important part of today systems and they deserve a first class mechanism in modeling languages, specially UML.

Following that premise, this paper presented an extension to UML activity diagrams centered on a construction for defining atomic composition of actions and activities. We have extended the UML notation with a particular stereotype for composite states and defined its semantics by the means of refinement morphisms over nonsequential automata.

Different lines of research are related to this work, manly semantics based on Abstract Machines, Petri nets and Workflow Statecharts. In [4] abstract state machines are used to provide semantics for activity diagrams in UML; [19] uses a process-like algebra for describing activity diagrams; and extensive work has been put on defining semantics for modeling workflow in [7, 6].

Some aspects of activity diagrams have been left out of this presentation and need further investigation. The most important are events and its associated action semantics.

Acknowledgment

This work was partially supported by CNPq (Project HoVer-CAM, GRAPHIT, E-Automaton) and FINEP/CNPq (Project Hyper-Seed) in Brazil.

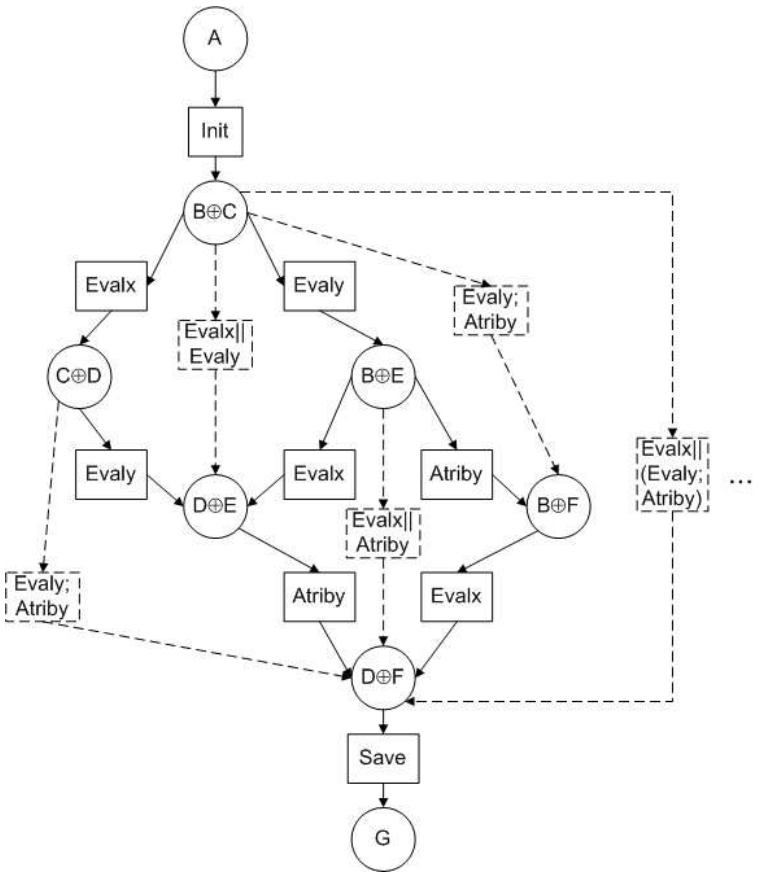


Figure 10: Nonsequential automata for activity diagram

References

[1] C. André and J. P. Rigault. Variations on the semantics of graphical models for reactive systems. In *System Management and Cybernetics*. IEEE Press, 2002.

[2] A. Asperti and G. Longo. *Categories, Types and Structures: an introduction to category theory for the working computer scientist*. MIT Press, Cambridge, 1990.

[3] G. Booch, J. Rumbaugh, and I. Jacobson. *The Unified Modeling Language User Guide*. Addison-Wesley, 1998.

[4] E. Borger, A. Cavarra, and E. Riccobene. An ASM semantics for UML activity diagrams. In *Lecture Notes in Computer Science - 8th International Conference on Algebraic Methodology and Software Technology*, volume 1816, pages 293–308. Springer-Verlag, 2000.

[5] W. Brauer, R. Gold, and W. Vogler. A survey of behaviour and equivalence preserving refinements of petri nets. In *Lecture Notes in Computer Science - Advances in Petri Nets 1990*, volume 483, pages 1–46. Springer-Verlag, 1991.

[6] R. Eshuis. *Semantics and Verification of UML Activity Diagrams for Workflow Modelling*. PhD thesis, University of Twente, 2002.

[7] R. Eshuis and R. Wieringa. Comparing petri net and activity diagram variants for workflow modelling - a quest for reactive petri nets. In *Lecture Notes in Computer Science - Petri Net Technology for Communication Based Systems*, volume 2472, pages 321–351. Springer-Verlag, 2003.

[8] M. Fowler and K. Scott. *UML Distilled - A Brief Guide to the Standard Object Modeling Language*. Addison-Wesley, 2000.

[9] T. Gehrke, U. Goltz, and H. Wehrheim. The dynamic models of UML: Towards a semantics and its application in the development process. Technical Report 11/98, Institut fur Informatik, Universitat Hildesheim, 1998.

[10] Object Management Group. *Unified Modeling Language Specification, version 1.5*. Rational Software Corporation, 2003.

[11] P. Blauth Menezes and J. F. Costa. Compositional reification of concurrent systems. *Journal of the Brazilian Computer Society*, 2(1):50–67, 1995.

[12] P. Blauth Menezes and J. F. Costa. Synchronization in petri nets. *Fundamenta Informaticae*, 26(1):11–22, 1996.

[13] P. Blauth Menezes, J. F. Costa, and A. S. Sernadas. Refinement mapping for general (discrete event) system theory. In *Lecture Notes in Computer Science - 5th International Conference on Computer Aided Systems Theory and Technology*, volume 1030, pages 103–116. Springer-Verlag, 1996.

[14] P. Blauth Menezes, J. P. Machado, and S. A. da Costa. Explicit and implicit nondeterministic refinement for concurrent, interacting systems. In *PDPTA2002 International Conference on Parallel and Distributed Processing Techniques and Applications*, Las Vegas, 2002. CSREA.

[15] P. Blauth Menezes, A. S. Sernadas, and J. F. Costa. Nonsequential automata semantics for concurrent, object-based language. In *Electronic Notes in Theoretical Computer Science - 2nd US-Brazil Joint Workshops on the Formal Foundations of Software Systems*, volume 14. Elsevier, 1998.

[16] J. Meseguer and U. Montanari. Petri nets are monoids. *Information and Computation*, 88(2):105–155, oct 1990.

[17] W. Reisig. *Petri Nets: an introduction*, volume 4 of *Eatcs Monographs on Theoretical Computer Science*. Springer-Verlag, 1985.

[18] W. Reisig and G. Rozenberg, editors. *Lectures on Petri nets: advances in Petri nets*. Springer-Verlag, 1998. Lecture Notes in Computer Science 1491.

[19] R. W. S. Rodrigues. Formalising UML activity diagrams using finite state processes. In *Workshop on Dynamic Behaviour in UML Models, 3rd International Conference on the Unified Modeling Language*, 2000.

[20] J. Rumbaugh, I. Jacobson, and G. Booch. *The Unified Modeling Language Reference Manual*. Addison-Wesley, 1999.

[21] G. Winskel and M. Nielsen. *Handbook of Logic in Computer Science*, volume 4, chapter Models for Concurrency, pages 1–148. Oxford University Press, 1995.

Integração de Fontes de Dados Heterogêneas Baseadas em Ambientes Flexíveis e Dinâmicos

Angelo Brayner
Universidade de Fortaleza (UNIFOR), Departamento de Ciência da Computação,
Fortaleza, Brasil, 60811-341,
brayner@unifor.br

e

Marcelo Meirelles
Universidade de Fortaleza (UNIFOR), Departamento de Ciência da Computação,
Fortaleza, Brasil, 60811-341,
meirelles.mia@unifor.br

Resumo
Ambientes flexíveis e dinâmicos são caracterizados pela alta independência dos participantes da conexão, pelo baixo controle sobre os serviços solicitados e disponibilizados e pela necessidade de uma alta tolerância às falhas de comunicação. Integrar fontes de dados baseadas nesses ambientes requer uma estratégia de integração que garanta uma maior autonomia para as fontes de dados locais. Por isso, esse trabalho propõe a utilização da arquitetura **MDBS** (Multidatabase System) para integrar fontes de dados heterogêneas disponibilizadas em ambientes flexíveis e dinâmicos. Na arquitetura MDBS, a linguagem de consultas é responsável por mapear e resolver todos os conflitos de integração e, portanto, deve possuir instruções especiais que permitam identificar tais conflitos. Conseqüentemente, esse artigo propõe, ainda, uma extensão à linguagem XQuery, denominada **MXQuery**, que apresenta suporte necessário à especificação de consultas que acessam múltiplas fontes de dados heterogêneas e distribuídas baseadas em um modelo de dados XML. Assim, a MXQuery pode ser incorporado a um MDBS para integrar fontes de dados heterogêneas.
Palavras chaves: Banco de Dados, Banco de Dados Múltiplos, Integração de Fontes de Dados, Fontes de Dados Heterogêneas, Linguagens de Consulta.

Abstract
Flexible and dynamic environments are characterized by high independence from connection participants, low control over available services and high tolerance to communication failures. Integrating data sources published on such environments requires an integration strategy that guarantees autonomy to the local data sources. Multidatabase Systems (**MDBS**) has been consolidated as an approach to integrate multiple heterogeneous and distributed data sources. A key property of MDBSs is to guarantee a higher autonomy to the local data sources than the other approaches for integrating heterogeneous data sources. MDBS technology uses a query language as integration mechanism, which is responsible for solving the integration conflicts. Thus, the query language must provide constructs to perform queries over several different data sources and capable of solving integration conflicts. This paper proposes an extension to the XQuery language, called **MXQuery**. The key feature of the proposed language is to provide mechanisms, which support the capability to jointly manipulate data in different data sources based on an XML data model.
Keywords: Databases, Multidatabases, Data Sources Integration, Heterogeneous Data Sources, Query Languages.

1. Introdução

Cada vez mais, a **Web** (*World Wide Web*) tem sido utilizada como um ambiente para disponibilizar bancos de dados, porém utilizar essa arquitetura impõe ao usuário as vantagens e desvantagens desse ambiente. A Web pode ser vista como uma grande rede constituída a partir da união de várias redes locais (**LAN** – *Local Area Network*) espalhadas pelo mundo inteiro, dessa forma, é possível acessar um banco de dados localizado em uma rede local distinta através da malha de rede formada pelo ambiente Web. Contudo, as redes locais, que integram a Web, são independentes entre si, podendo conectar-se ou desconectar-se da rede mundial a qualquer momento.

Recentemente, alguns trabalhos têm abordado a conexão de redes locais sem fio e de forma dinâmica, essas redes são conhecidas como **MANET** (*Mobile Ad hoc Networking*) [4]. As redes MANET possuem um raio de cobertura e os equipamentos que entram nesse raio de cobertura passam automaticamente a fazer parte da rede estabelecida. Portanto, equipamentos móveis podem trafegar por diversas áreas de cobertura de redes MANET distintas e, embora, esse ambiente esteja limitado a uma pequena área de abrangência, ele possui algumas características semelhantes ao ambiente Web, tais como: alta independência dos participantes da conexão; baixo controle sobre os serviços solicitados e disponibilizados; alta tolerância às falhas de comunicação.

Nesse sentido, pode-se vislumbrar um cenário, onde vários bancos de dados podem estar inteconectados através de um sistema de comunicação sem fio. Dessa forma, um usuário, a partir de um computador móvel qualquer, poderia acessar quaisquer bancos de dados, residindo também em computadores móveis, desde que os computadores móveis estivessem localizados em uma região coberta por um sistema de comunicação sem fio. Uma coleção de computadores móveis interconectados por uma infra-estrutura de comunicação sem fio é denominada de comunidade de bancos de dados móveis (**MDBC** – *Mobile Database Comunity*), onde cada participante desta comunidade pode atuar como um servidor de banco de dados autônomo [4].

Portanto, integrar bancos de dados disponibilizados na Web e em redes MANET tem se tornado um grande desafio para área de banco de dados. Além disso, deve-se considerar a possibilidade desses bancos de dados serem heterogêneos, onde, nesse caso, para fornecer uma resposta consolidada ao usuário, é necessário eliminar as diferenças existentes (resolução de conflitos) entre os bancos de dados heterogêneos participantes. Várias abordagens têm sido propostas para viabilizar essa integração, cada uma delas sendo mais adequada a uma determinada situação. Por esse motivo, a definição da estratégia de integração mais apropriada deve considerar as características dos bancos de dados a serem integrados e o contexto em que a integração é necessária.

Uma abordagem para integração de bancos de dados, que estejam baseados na Web ou em redes *ad hoc*, deve oferecer suporte às características desses ambientes. Portanto, a arquitetura de integração deve conferir um alto grau de autonomia para os participantes. Além disso, deve ser capaz de resolver conflitos de integração originados, principalmente, da grande heterogeneidade dos bancos de dados a serem integrados. Finalmente, deve permitir uma manipulação de dados distribuídos em diferentes locais de forma eficiente.

Uma estratégia para integrar múltiplos bancos de dados heterogêneos é a de Sistema de Bancos de Dados Múltiplos (**MDBS** – *Multidatabase System*). A tecnologia de MDBS garante um maior grau de autonomia para os bancos de dados participantes da integração, denominados de Sistemas de Bancos de Dados Locais (**LDBS** – *Local Database System*) [8, 16, 6]. Entretanto, utilizar essa abordagem obriga o usuário a definir questões de localização e resolução de problemas de integração, como, por exemplo, conflitos semânticos e estruturais entre os dados armazenados nos diversos LDBS's. Em um MDBS, não é necessária a definição de um esquema global de integração, pois a linguagem de consulta (**MDL** – *Multidatabase Language*) é utilizada como mecanismo de integração. A idéia é que os usuários sejam capazes de "enxergar" cada esquema dos LDBS's representados em um modelo comum de dados (**CDM** – *Common Data Model*) [8]. Os esquemas locais representados nesse modelo comum são denominados de esquema conceitual.

A idéia básica desse trabalho é utilizar o **XML** (*eXtended Markup Language*) como modelo comum (esquema conceitual) dos dados armazenados nas múltiplas fontes de dados heterogêneas. Conseqüentemente, para adotar a arquitetura MDBS, deve-se utilizar uma linguagem de consultas MDL que seja capaz de manipular bancos de dados múltiplos de forma integrada e baseada em um modelo de dados XML. Assim, foi proposta a linguagem MXQuery que é uma extensão da linguagem XQuery (*Working-in-progress*). A MXQuery resolve problemas de integração de dados como heterogeneidade semântica e o tratamento de informações incompletas. A estrutura da extensão é bastante flexível, pois garante a integração de um número variável de fontes de dados com diferentes graus de autonomia.

Além disso, esse trabalho apresenta uma arquitetura para o processamento de consultas em sistemas de banco de dados múltiplos. A idéia é que a linguagem MXQuery seja incorporada a esta arquitetura como linguagem de banco de dados múltiplos.

O resto desse artigo está organizado como se segue. Uma breve apresentação de trabalhos relacionados é feita na seção 2. A proposta de extensão a linguagem XQuery é apresentada na seção 3. A seção 4 apresenta mais detalhadamente as características da estratégia de integração proposta. A seção 5 discute o processamento de consultas MXQuery. Conclusões e trabalhos futuros aparecem na seção 6.

2. Trabalhos Relacionados

Inúmeros trabalhos apresentam linguagens de consultas para bancos de dados onde é possível manipular fontes de dados distintas. Em [12, 10], é apresentada uma linguagem MDL para manipulação de bancos de dados múltiplos. A linguagem MSQL possui mecanismos para integrar bancos de dados relacionais, que aderem ao padrão SQL de forma a permitir junções entre tais bancos de dados. Embora essa proposta confira maior expressividade à linguagem SQL, ela é limitada no tratamento de incompatibilidades estruturais. Apenas conflitos de nome (sinônimos e homônimos) são resolvidos através da utilização de variáveis semânticas e a junção entre bancos de dados é limitada a operações com atributos de domínios compatíveis [14]. Além disso, Krishnamurthy et al. apontam a incapacidade da linguagem MSQL em resolver discrepâncias esquemáticas ou conflitos de esquema [11].

Outro fator limitante da linguagem MSQL é permitir apenas a integração de esquemas relacionais. Conseqüentemente, para utilizar essa linguagem, seria necessário adotar como CDM o modelo de dados relacional. A tarefa de mapear modelos de dados pós-relacionais, em especial os baseados em dados semi-estruturados, para um modelo de dados tão rígido quanto o modelo de dados relacional é uma tarefa bastante complexa. Portanto, a MDL MSQL não é indicada para integrar fontes de dados baseadas na Web ou em redes *ad hoc*.

A proposta apresentada por Missier e Rusinkiewiez define atributos globais, que representarão o resultado integrado de uma consulta [14]. Em seguida, utilizam instruções declarativas de mapeamento entre os atributos dos bancos de dados locais e os atributos globais. Após a definição dos mapeamentos ou contexto de uma consulta, as consultas, que são expressas em uma linguagem derivada do SQL, podem ser submetidas. A utilização de junções implícitas possibilita expressar consultas mais flexíveis e orientadas ao resultado.

A linguagem IDL apresentada por Krishnamurthy et al. permite endereçar os problemas abordados pela linguagem MSQL, porém a principal característica da linguagem IDL é resolver problemas de conflito de esquemas, denominados de discrepâncias esquemáticas [11]. Embora parte desses conflitos possa ser resolvida através da linguagem MSQL, alguns problemas não encontram em MSQL uma solução, como, por exemplo, integrar informações representadas como dado em um esquema e metadado em outro esquema. Tanto a proposta apresentada por Krishnamurthy et al. [11] quanto à apresentada por Missier e Rusinkiewiez [14] possuem mais expressividade comparada à linguagem MSQL, contudo continuam limitadas pelo modelo de dados relacional.

Domenig e Dittrich apresentaram uma linguagem baseada em OQL, onde os dados podem ser visualizados em diferentes níveis de visões conceituais. Tais níveis de visões conceituais organizam os dados de fontes de dados distintas de acordo com características comuns. Esse tipo de apresentação foi chamado pelos autores de *data space*, que é utilizado para integrar dados estruturados, semi-estruturados e não estruturados. Apesar da linguagem OQL apresentar mecanismos para consultar conjuntos de dados, esses mecanismos não podem ser utilizados para consultar dados não estruturados e semi-estruturados. Por esse motivo, os autores apresentaram uma extensão à linguagem OQL, denominada SOQL [6].

A abordagem proposta pelos autores define duas etapas distintas para integração de fontes de dados. Na etapa de pré-integração, os administradores identificam similaridades e conflitos entre as fontes de dados e, em seguida, definem regras de correspondência entre esquemas (contexto da consulta). Na etapa de integração, o esquema integrado é automaticamente construído com base nas regras previamente estabelecidas. Embora a abordagem proposta pelos autores ofereça um certo grau de flexibilidade com a utilização das regras de correspondência, esse processo tem que ser executado antes da proposição de consultas. Além disso, é necessário redefinir o esquema integrado cada vez que uma regra é modificada. Caso a quantidade de regras de correspondência seja muito grande pode degradar a performance do sistema.

A proposta apresentada por Sattler et al. assemelha-se à estratégia adotada por [14]. A linguagem FRAQL possui instruções declarativas que possibilitam mapear os objetos de um esquema local em objetos de um esquema global [18]. Inicialmente, o esquema global, que estará acessível às consultas dos usuários, é definido. Em seguida, os esquemas locais são mapeados segundo o esquema global, onde é possível utilizar funções de conversão criadas pelos próprios usuários. As correspondências criadas pelo usuário ou pelo administrador formam o contexto de execução de consultas. As consultas são submetidas considerando o esquema global e, em seguida, são transformadas em sub-consultas utilizando os mesmos critérios estabelecidos no mapeamento.

Diferentemente da proposta adotada por [6], essa abordagem não necessita que o esquema global seja refeito cada vez que uma nova fonte de dados entre na comunidade, sendo necessário, apenas, que essa nova fonte de dados estabeleça sua correspondência com o esquema já definido. Embora essa proposta seja mais flexível, o fato de a etapa de mapeamento estar dissociada da consulta propriamente dita, torna a especificação de consultas menos dinâmica que a abordagem utilizada nesse artigo. Além disso, a abordagem proposta em [18] está direcionada ao modelo de dados objeto-relacional e, embora seja mais flexível que o modelo relacional para representar dados semi-estruturados, ainda necessita de algum esforço para representar fontes de dados semi-estruturadas e não estruturadas.

A linguagem XQuery [5, 17, 2] possui várias características que permitem consultar documentos XML e apresentar como resultado dessa consulta novos documentos XML. Os documentos XML resultados podem ser sub-árvores do documento XML original ou podem apresentar estrutura diferente, inclusive com a utilização de novos elementos obtidos através de construtores de elementos. Essas características tornam a linguagem bastante

expressiva e poderosa, sendo possível, inclusive, consultar múltiplas fontes de dados XML. Embora seja possível exprimir operações de união e junção entre essas fontes de dados, a linguagem não apresenta mecanismos para resolução de conflitos. Dessa forma, as consultas XQuery assumem que as fontes de dados, embora distintas, foram concebidas dentro de um único projeto.

A XQuery permite a construção de elementos, o que torna possível contornar a falta de alguns mecanismos para resolução de conflitos. Embora os construtores de elementos forneçam alguns mecanismos para contornar o problema da heterogeneidade semântica, não existe um mapeamento formal entre atributos de fontes de dados distintas e, portanto, tais construtores não expressam de forma clara o relacionamento conceitual existente entre as fontes de dados que estão sendo integradas. Conseqüentemente, as consultas XQuery tendem a ser mais complexas que o necessário. Além disso, a linguagem XQuery não considera problemas de conexão e desconexão de fontes de dados e essa característica é fundamental em ambientes dinâmicos.

3. A Linguagem MXQuery

A linguagem MXQuery é uma extensão da linguagem XQuery, onde foram acrescentadas características que permitem manipular e integrar fontes de dados heterogêneas, tendo XML como modelo de dados comum para representação de esquemas conceituais. A linguagem XQuery é, naturalmente, orientada a consultar dados semi-estruturados e, portanto, apresenta estruturas flexíveis para identificar as informações requeridas e arranjá-las de acordo com as especificações do usuário. A linguagem MXQuery aproveitou-se dessas características e implementou declarações de mapeamento entre fontes de dados distintas para representar o relacionamento conceitual existente entre essas fontes de dados.

A idéia por trás da linguagem MXQuery é utilizar declarações de mapeamento entre os elementos originais das fontes de dados e um ou mais elementos de um documento XML que conterá o resultado da consulta. O processo de mapeamento está inserido na própria consulta, tornando, dessa forma, a MXQuery mais flexível que as propostas apresentadas em [14] e [18]. Além disso, é possível utilizar variáveis de resultado (associadas aos elementos do documento XML resultado) e variáveis das fontes de dados (associadas aos elementos das fontes de dados) em especificações de condições na cláusula "*WHERE*". Isso permite que sejam inseridos filtros (condições) sobre dados integrados e sobre dados das fontes de dados locais.

Por esse motivo, as condições existentes na consulta são processadas em dois momentos distintos. As condições sobre dados das fontes de dados locais são inseridas nas sub-consultas encaminhadas às fontes de dados, limitando, portanto, o resultado da sub-consulta sobre essas fontes de dados. As condições sobre os dados integrados são processadas após a etapa de construção do resultado global, ou seja, elementos inicialmente inseridos no resultado podem ser excluídos ou modificados de acordo com as condições sobre os dados integrados.

Por exemplo, a expressão ***WHERE $p/livro/ano > "1999"***, onde "$p" representa um documento XML resultado, só será avaliada após a integração dos dados. Por isso, poderá ser gerado um tráfego desnecessário pela rede. Elementos "livro" que possuam sub-elementos "ano" menores que 1999 são enviados para o gerenciador de consultas e, só após a etapa de integração, esses elementos são descartados. Contudo, durante a etapa de simplificação, a arquitetura MDBS tenta transformar as condições baseadas em dados integrados em condições baseadas nas fontes de dados locais, pois essa simplificação reduz a quantidade de dados trafegada pela rede. Por exemplo, a condição acima deve ser reescrita pelo gerenciador de consultas, como ***WHERE $d1/livro/ano > "1999" AND $d3/book/year > "1999"***, onde "$d1" e "$d2" identificam as fontes de dados locais (ver consulta 2 – seção 3.2.).

Diferentemente das propostas [14], [6] e [18], a linguagem MXQuery não requer a definição de contextos de uma consulta. Por esse motivo, as consultas expressas através da linguagem MXQuery são mais dinâmicas e, conseqüentemente, sofrem menor interferência da evolução dos esquemas locais, conexão e desconexão de novas fontes de dados. Isso significa que, no momento em que uma consulta MXQuery é proposta, toda a informação sobre como integrar as fontes de dados consultadas é expressa na própria consulta. Dessa forma, a evolução dos esquemas locais não compromete as definições do ambiente de integração, apenas exige novas formas de expressar as consultas MXQuery.

Uma grande dificuldade em processar consultas, sobre múltiplas fontes de dados heterogêneas que merece destaque, é o tratamento da disponibilidade das fontes de dados a serem consultadas [20, 1]. A estratégia adotada pela linguagem MXQuery para resolver esse problema é especificar as diversas fontes de dados de uma consulta *C* e, em seguida, classificá-las de acordo com sua relevância para *C*. Se uma fonte de dados é imprescindível para a consulta *C*, então ela é classificada como obrigatória, caso contrário, ela é classificada como opcional. As fontes de dados opcionais, não disponíveis durante a execução de uma consulta, não interrompem o processamento dessa consulta. Com isso, fornece-se ao processador de consultas do MDBS a capacidade de alterar a consulta original para retirar qualquer referência a essa fonte de dados opcional. Em contrapartida, as fontes de dados classificadas como obrigatórias não podem estar indisponíveis no momento da consulta, caso isso ocorra, a consulta não é executada.

[42]	EFLWORExpr	::=	EachClause (ForClause | LetClause) * WhereClause? OrderByClause? "return" ExprSingle
[144]	EachClause	::=	"each" "$" VarName "full"? ("," "$" VarName "full"?)* DefClause
[145]	DefClause	::=	AliasClause (AliasClause)* (EqualClause)*
[146]	AliasClause	::=	"alias" PathExpr "$" VarName "null"? ("," "$" VarName "null"?)*
[147]	EqualClause	::=	"equal" RelativePathExpr* "is" RelativePathExpr "key"? "hide"?
[43]	ForClause	::=	"for" "$" VarName TypeDeclaration? PositionalVar? "in" ExprSingle ("," "$" VarName TypeDeclaration? PositionalVar? "in" ExprSingle)*
[45]	LetClause	::=	"let" "$" VarName TypeDeclaration? ":=" ExprSingle ("," "$" VarName TypeDeclaration? ":=" ExprSingle)*
[122]	TypeDeclaration	::=	"as" SequenceType
[44]	PositionalVar	::=	"at" "$" VarName
[46]	WhereCaluse	::=	"where" Expr
[47]	OrderByCaluse	::=	("order" "by" | "stable" "order" "by") OrderSpecList
[48]	OrderSpecList	::=	OrderSpec ("," OrderSpec)*
[49]	OrderSpec	::=	ExprSingle OrderModifier
[50]	OrderModifier	::=	("ascending" | "descending")? (("empty" "greatest") | ("empty" "least"))? ("collation" StringLiteral)?

Tabela 1. Sintaxe das Expressões EFLWOR.

3.1. Expressões EFLWOR

A idéia básica da MXQuery é estender a estrutura FLWOR presente na linguagem XQuery. Na MXQuery poderão existir expressões do tipo **EFLWOR** (*Each* – FLWOR), além de expressões FLWOR. Uma consulta baseada em uma expressão EFLWOR percorre todos os elementos das árvores referentes aos documentos XML que representam as fontes de dados participantes da consulta. O resultado da consulta é uma árvore, que é denominada árvore resultado. A árvore resultado é construída através da união, junção e/ou fusão de elementos pertencentes a documentos (fontes de dados) distintos.

Por se tratar de uma extensão da linguagem XQuery, parte da gramática da linguagem MXQuery já está descrita em [2]. A tabela 1 descreve a sintaxe das expressões EFLWOR onde a notação EBNF foi utilizada. A primeira coluna da tabela na gramática original apresenta um número seqüencial para todos as cláusulas definidas pela gramática. A tabela 1 inseriu elementos numerados de 144 a 147. O elemento 42 deve substituir o elemento da gramática XQuery original e os outros elementos presentes na tabela 1 foram repetidos por uma questão de clareza.

As expressões EFLWOR introduzidas pala linguagem MXQuery originaram cláusulas não definidas para a linguagem XQuery. Tais cláusulas são as seguintes: "EACH", "ALIAS", "EQUAL", "FULL", "NULL", "HIDE" e "KEY". Essas cláusulas são utilizadas para identificar as fontes de dados consultadas, especificar mapeamentos entre atributos locais (originados das fontes de dados) e atributos globais (representantes dos documentos resultado).

EACH. Essa cláusula associa valores a uma ou mais variáveis, denominadas variáveis de resultado. Essas variáveis são utilizadas para a construção de um resultado que poderá representar, na realidade, elementos XML gerados a partir da integração das fontes de dados participantes do mecanismo de integração. Por exemplo, a declaração ***EACH $p*** define que uma variável "$p" conterá o resultado de um processo de integração de elementos pertencentes a fontes de dados distintas. Regras para o processo de integração (por exemplo, regras para a resolução de conflitos) deverão ser definidas nas demais declarações da linguagem MXQuery. A variável declarada na cláusula "EACH" é um documento XML bem formado e pode ser referenciada por qualquer declaração da estrutura EFLWOR que aceite como parâmetro uma árvore de nós.

ALIAS. Essa cláusula especifica as árvores (ou sub-árvores) XML que serão utilizadas na consulta onde cada árvore referenciada pode representar uma fonte de dados distinta. Portanto, essa cláusula é utilizada para definir o escopo da consulta. Um documento referenciado na declaração "ALIAS" é associado a uma ou mais variáveis, denominadas variáveis de fonte de dados. De forma análoga às variáveis definidas pela cláusula "EACH", essas variáveis podem ser utilizadas nas cláusulas que aceitem uma árvore de nós como parâmetro. Por exemplo, a declaração ***ALIAS document("d1.xml")/bib $d1*** define uma variável "$d1" que representa a fonte de dados especificada pelo documento "d1.xml"; a função "document" definida em [2] identifica o documento XML que

representa uma das fontes de dados participantes da consulta, além de indicar o ponto de entrada na floresta de nós do documento XML. Portanto, uma variável definida através da cláusula "ALIAS" pode representar qualquer sub-árvore da fonte de dados referenciada. A ordem de declaração de cláusulas "ALIAS" em uma consulta MXQuery especifica a ordem de prioridade entre as fontes de dados a serem acessadas. Dessa forma, a fonte de dados, especificada na primeira declaração "ALIAS", é a fonte de dados com maior ordem de prioridade e assim por diante. Em casos de conflito de dados, a ordem de prioridade será utilizada para resolver esse tipo de conflito segundo o seguinte critério: serão considerados válidos os valores de elementos pertencentes a fontes de dados com maior ordem de prioridade.

EQUAL. Essa cláusula especifica as regras de integração de elementos de documentos distintos onde grande parte dos conflitos é resolvida. A construção identifica a relação entre os elementos de documentos distintos e, em seguida, um novo elemento contendo o resultado da integração é gerado. Por exemplo, na especificação ***EQUAL $d1/livro $d3/book IS $p/livro***, um elemento (no caso, livro) é inserido na árvore resultado. A função desse novo elemento é integrar os elementos "livro" e "book" pertencentes a fontes de dados distintas. Nesse exemplo, os dois elementos originais são especificados e caracterizados, portanto, como sinônimos. Contudo essa especificação pode ser abreviada através da omissão do elemento da fonte de dados local identificada pela variável "$d1". Essa abreviação é possível porque, em uma declaração de cláusulas "ALIAS", existe um documento (no caso $d1) que possui um elemento "livro". A declaração abreviada seria ***EQUAL $d3/book IS $p/livro***. Na consulta exemplo (figura 1) pode ser observada a utilização do mecanismo de abreviação em uma cláusula "EQUAL". Normalmente, apenas os elementos referenciados em alguma declaração da cláusula "EQUAL" comporão a variável especificada na cláusula "EACH" (veja definição da cláusula "FULL").

FULL. A cláusula opcional "FULL" pode ser utilizada em associação com a cláusula "EACH". Essa cláusula especifica, que todos os elementos de todas as fontes de dados declaradas na cláusula "ALIAS" comporão a árvore resultado associada à variável de resultado especificada na cláusula "EACH", mesmo que não sejam referenciados na cláusula "EQUAL". Se nenhuma forma de restrição for especificada, esse tipo de declaração ocasiona a UNIÃO de todas as fontes de dados referenciadas. Diferentemente do modelo de dados estruturados (bancos de dados convencionais), onde a operação de união necessita que as fontes de dados sejam compatíveis, documentos XML não oferecem nenhuma restrição para operações de união.

NULL. A cláusula opcional "NULL" pode ser utilizada em conjunto com a cláusula "ALIAS". Essa cláusula especifica que a fonte de dados ou o documento referenciado não é imprescindível para a consulta. Dessa forma, caso a fonte de dados não esteja acessível ou não seja declarada corretamente, qualquer declaração referente a essa fonte de dados (ou algum elemento pertencente à mesma) será ignorada. Por exemplo, se uma consulta especificar a declaração ***ALIAS document("d3.xml")/lib $d3 NULL***, então o documento "d3.xml" não será considerado imprescindível para o resultado dessa consulta. Nesse caso, mesmo que a fonte de dados, contendo o documento, não esteja acessível, a consulta será processada. Da mesma forma, os elementos pertencentes ao documento "d3.xml" que aparecerem em outras declarações serão suprimidos. A declaração ***EQUAL $d1/livro $d3/book IS $p/livro*** é processada como se o elemento "$d3/book" não fizesse parte da expressão. A supressão do elemento "$d3/book" também ocorreria caso esse elemento fosse declarado incorretamente. Isso possibilita que alterações nos esquemas das fontes de dados não inviabilizem uma consulta. Esta propriedade é fundamental para ambientes como a Web e redes *ad hoc*.

HIDE. A cláusula opcional "HIDE" pode ser utilizada em conjunto com a cláusula "EQUAL". A cláusula indica que o resultado da relação declarada através da cláusula "EQUAL", mesmo compondo a variável especificada na cláusula "EACH", não deve aparecer na árvore resultado apresentada pela cláusula "RETURN". Por exemplo, na declaração ***EQUAL $d3/book/year $d1/livro/ano IS $p/livro/ano HIDE*** o elemento "livro/ano" (resultado da resolução de um conflito de sinônimos) pode ser referenciado em outras cláusulas da consulta, no entanto, ele não faz parte dos elementos apresentados pela cláusula "RETURN". O elemento "livro/ano" é um elemento virtual e seu ciclo de vida termina após a avaliação de todas as expressões condicionais presentes na consulta. A cláusula "HIDE" tem precedência sobre a cláusula "FULL", ou seja, um elemento declarado que utilize a cláusula "HIDE" não comporá a árvore resultado apresentada, mesmo tendo sido utilizada a cláusula "FULL".

KEY. A cláusula opcional "KEY" pode ser utilizada em conjunto com a cláusula "EQUAL", onde indica que os elementos especificados na declaração devem servir como identificador de equivalência entre objetos. Algumas consultas podem determinar que objetos que representem o mesmo conceito do mundo real devem sofrer uma fusão, onde uma única representação do objeto deve existir. Contudo, é necessário especificar quando dois objetos são idênticos. Isso é feito através da comparação entre propriedades de fontes de dados distintas que são indicadas pela cláusula "KEY". Considerando a declaração ***EQUAL $d1/livro/@isbn $d3/book/isbn IS $p/livro/isbn KEY***, os documentos "d1.xml" e "d3.xml" são processados originando um novo elemento "livro/isbn" que integra os elementos das fontes de dados originais. O conteúdo das fontes de dados sofrerá uma união para formar a árvore

resultado, no entanto, caso o valor associado a algum elemento "$d1/livro/@isbn" seja igual ao valor de algum elemento "$d3/book/isbn", então esses dois objetos (considerados idênticos) sofrerão uma fusão.

3.2. Exemplo de uma Consulta MXQuery

A utilidade e aplicabilidade da linguagem MXQuery serão ilustradas nesta seção. A idéia é especificar uma consulta, de acordo com o seguinte cenário. Considere que algumas livrarias resolvem fundar um consórcio, onde seus títulos seriam disponibilizados na Web. Cada livraria gerencia sua própria fonte de dados. Muito provavelmente, tais fontes de dados são heterogêneas. Contudo, essa parceria entre as livrarias requer que os dados referentes aos seus acervos sejam "vistos" de forma integrada. Porém, nenhuma das livrarias deseja perder o controle sobre seus dados locais. Dessa forma, devem-se integrar fontes de dados heterogêneas, preservando a autonomia local de cada uma delas.

Suponha que a integração dos dados das diversas livrarias é realizada através da abordagem de MDBS e considere que os esquemas conceituais correspondentes a cada fonte de dados local serão representados em XML. Por uma questão de simplicidade, em apêndice A são representadas duas fontes de dados "d1.xml" e "d3.xml" com seus respectivos conteúdos. Entretanto, a arquitetura MDBS disponibiliza de fato documentos XML *Schema* que representam o mapeamento dos esquemas das fontes de dados locais em documentos representados no modelo de dados XML.

Consulta 1: Retornar títulos, autores e preços dos livros existentes nas livrarias, cujos esquemas conceituais estão representados pelos documentos "d1.xml" e "d3.xml". Utilize o ISBN para especificar a equivalência entre os objetos de fontes de dados distintas. Considere ainda que os preços representados nos documentos em questão estão expressos em moedas distintas, onde U$ 1,00 equivale à R$ 2,98.

Uma possível expressão MXQuery é apresentada na figura 1. A execução da consulta retornará uma variável "$p" contendo elementos "livro" dos documentos d1.xml e d3.xml (especificados nas cláusulas "ALIAS"), cada um contendo os sub-elementos "titulo", "autor" e "preco". Na linha para recuperar a informação de preço, é necessário efetuar uma conversão de moeda (neste exemplo de Dólar para Real). Essa função de conversão resolve, portanto, um conflito de domínio existente entre as duas fontes de dados especificadas.

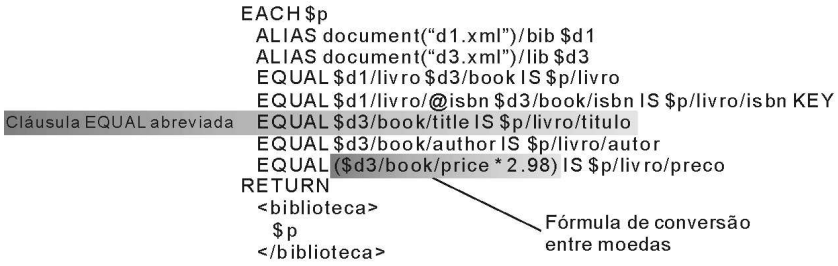


Figura 1. Consulta 1 Expressa em MXQuery.

Observe que a cláusula "KEY" especifica a propriedade que deve ser utilizada para identificar objetos idênticos, onde objetos considerados idênticos devem sofrer uma fusão no processo de integração. Ainda nessa declaração, pode ser identificada a resolução de um conflito de esquema, visto que se pode comparar diretamente o atributo "livro/@isbn" com o sub-elemento "book/isbn". Essa declaração insere o sub-elemento "isbn" na árvore resultado, como resultado da integração. Observe que as declarações de equivalência de título, autor e preço estão na sua forma abreviada, contudo, esse fato não causa nenhum impacto no resultado da consulta.

Consulta 2: Retornar os títulos dos livros existentes nas livrarias cujos esquemas conceituais estão representados pelos documentos XML "d1.xml" e "d3.xml". Utilize o ISBN para suprimir duplicidades. Os livros devem ter sido publicados após 1999.

A expressão MXQuery mostrada na Figura 2 pode ser utilizada para expressar esta consulta. A consulta 2 é similar à consulta 1, porém foi acrescentada a declaração de equivalência do sub-elemento "ano" com "year", resolvendo dessa forma um conflito de sinônimos. Observe que, diferentemente das declarações de equivalência de sub-elementos especificadas na consulta anterior, a declaração especificada na consulta 2 não determina que o sub-elemento "ano" componha o resultado dessa consulta, devido a utilização da cláusula "HIDE". A idéia aqui é utilizar esse sub-elemento (após a resolução do conflito de sinônimo) apenas para o filtro especificado na cláusula "WHERE".

Conforme os documentos apresentados no apêndice A, existe um conflito de dados entre dois dos elementos que comporão a árvore resultado dessa consulta. Dois elementos "livro" que apresentam o mesmo ISBN possuem o sub-elemento "ano" e "titulo" com valores distintos, onde o sub-elemento "ano" possui um dos valores igual a 2001 e o outro igual a 1999. No caso do sub-elemento "titulo", o conteúdo de sub-elemento é apresentado em português no documento "d1.xml" e em inglês no documento "d3.xml".

```
EACH $p
  ALIAS document("d1.xml")/bib $d1
  ALIAS document("d3.xml")/lib $d3 NULL
  EQUAL $d3/book IS $p/livro
  EQUAL $d1/livro/@isbn $d3/book/isbn IS $p/livro/@isbn KEY
  EQUAL $d3/book/title IS $p/livro/titulo
  EQUAL $d3/book/year IS $p/livro/ano HIDE
WHERE $p/livro/ano > "1999"
RETURN
  <biblioteca>
    $p
  </biblioteca>
```

Condição baseada em variável de resultado

Figura 2. Consulta 2 Expressa em MXQuery.

Esse conflito de dados é resolvido através da indicação da fonte de dados cujo valor de seus elementos deverá prevalecer. Essa indicação é feita através da ordem das declarações das fontes de dados consultadas. Na consulta 2, a fonte de dados especificada pela variável "$d1" é priorizada, significando que, em caso de conflito de dados o seu conteúdo será preservado. Devido a esse fato, o resultado da integração entre as fontes de dados "d1.xml" e "d3.xml" determina que o valor do sub-elemento "ano" deve ser igual a 2001 e o título em português deve constar no resultado. Nesse exemplo, se a ordem das fontes de dados na cláusula "ALIAS" fosse invertida, o resultado da consulta seria um elemento "livro" vazio. Observe que, na consulta 1, também ocorreram conflitos de dados (sub-elementos "titulo" e "preço"), que foram resolvidos da mesma forma.

A utilização da cláusula "NULL" na consulta determina que a indisponibilidade do documento "d3.xml" durante a execução da mesma não inviabiliza o resultado. Coincidentemente, o resultado apresentado é o mesmo, independente da disponibilidade de "d3.xml", no entanto, sua indisponibilidade ocasionaria um erro se a cláusula "NULL" não fosse especificada.

A condição utilizada na cláusula "WHERE" da segunda consulta é baseada em uma variável de resultado. Isso significa que, a princípio, para que essa condição seja avaliada, é necessário realizar a integração entre as duas fontes de dados e, em seguida, verificar a condição. Contudo, no processo de otimização de consultas, essa condição pode ser substituída por uma condição baseada em fonte de dados. A expressão equivalente seria: ***WHERE $d1/livro/ano > "1999" AND $d3/book/year > "1999"***. Pode-se notar que caso o processo de otimização execute essa substituição, então a fonte de dados "d3.xml" não enviará nenhuma resultado parcial (eliminando tráfego de rede desnecessário). Além disso, nenhum conflito de dados ocorrerá, pois apenas o documento "d1.xml" resultará um elemento cujo ISBN será igual a "352".

4. Estratégia de Integração

Esse trabalho propõe utilizar a tecnologia de MDBS como a estratégia de integração de fontes de dados heterogêneas. A Figura 3 apresenta um modelo abstrato da arquitetura utilizada para integrar dados armazenados em fontes de dados heterogêneas, distribuídas e autônomas.

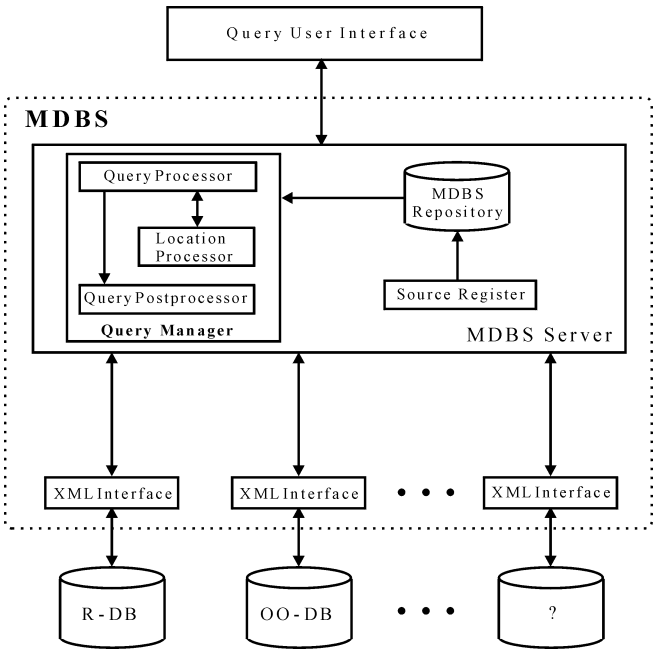


Figura 3. Arquitetura do Mecanismo de Integração.

Na Figura 3, a interface de consulta de usuário (**QUI** – *Query User Interface*) faz a intermediação entre o usuário e a arquitetura de integração, sendo de responsabilidade desse componente receber as consultas e apresentar os resultados. Um usuário, que desejar acessar múltiplas fontes de dados integradas através da arquitetura, deverá enviar ao QUI uma consulta MXQuery, denominada de consulta global. Pode-se, ainda, pensar na QUI como um agente móvel que transporta consultas MXQuery para serem executadas pela arquitetura de integração [4]. As consultas globais são enviadas, então, ao gerenciador de consultas (**QM** – *Query Manager*). A principal funcionalidade deste componente é dividir uma consulta global em sub-consultas e, em seguida, enviá-las às fontes de dados correspondentes.

O gerenciador de consulta está dividido, por sua vez, em três componentes principais, que são: processador de consulta (**QP** – *Query Processor*), processador de localização (**LP** – *Location Processor*) e pós-processador de consulta (**QPp** – *Query Postprocessor*). O processador de consulta deve receber as consultas expressas em MXQuery, identificar as fontes de dados (ou interfaces das fontes de dados) referenciadas, simplificar e otimizar

as consultas globais, gerar as sub-consultas e finalmente, encaminhar as sub-consultas para as fontes de dados correspondentes.

O processador de localização desempenha um papel auxiliar para o processador de consulta, pois ele deve localizar as fontes de dados referenciadas e indicar erros de localização. Os erros de localização podem significar a indisponibilidade de alguma fonte de dados referenciada. Essas informações são úteis no processo de simplificação de consultas, pois as fontes de dados inacessíveis devem ser excluídas da consulta. O processo de exclusão de uma fonte de dados de uma consulta deve obedecer às regras estabelecidas na própria consulta. Dessa forma, nem sempre será possível excluir a fonte de dados da consulta, nesse caso, a consulta inteira é rejeitada. Caso seja possível excluir a fonte de dados, então todas as expressões que referenciam a fonte de dados excluída devem ser excluídas. As fontes de dados que podem ser excluídas são identificadas na sintaxe do MXQuery, através da utilização da cláusula "NULL". O pós-processador de consulta será explicado mais à frente.

As sub-consultas são enviadas para as interfaces XML (*XML Interface*), onde são traduzidas para linguagem de consulta nativa e direcionadas para as fontes de dados locais correspondentes às interfaces XML. Portanto, há uma interface XML associada a cada fonte de dados local. O papel desempenhado pela interface XML será discutido posteriormente.

Depois de processada localmente, o resultado de uma sub-consulta é enviado à interface XML, onde novamente é traduzido (para XML) e encaminhado ao pós-processador de consulta. Esse componente deve utilizar as regras estabelecidas na consulta original para resolução de conflitos. Por exemplo, pode-se estabelecer uma regra para resolver problemas de sinônimo utilizando a expressão "EQUAL". Portanto, através da MXQuery, o usuário é capaz de fornecer informação para que o pós-processador de consulta resolva conflitos.

Após a resolução dos conflitos, o QPp deve combinar o resultado das diversas sub-consultas recebidas, de acordo, com as regras de construção de resultados também estabelecidas na consulta original. O resultado combinado das sub-consultas é, finalmente, encaminhado à interface de consulta de usuário.

Dois outros componentes são necessários para o funcionamento da estratégia proposta. O registrador de fonte (**SR** – *Source Register*) é encarregado de controlar a conexão e desconexão das fontes de dados participantes do MDBS. Esse componente recebe as requisições de participação no MDBS enviadas pelas fontes de dados locais, registra a localização física das fontes de dados (**URL** – *Uniform Resource Location*) e publica o esquema disponibilizado pelas fontes de dados locais.

O repositório MDBS (*MDBS Repository*) deve armazenar as informações do MDBS. Essas informações incluem os meta-dados das fontes de dados participantes, definições do MDBS, como por exemplo, as fontes de dados participantes ou consultas pré-formatadas, e localização física das fontes de dados participantes.

Uma interface XML é responsável por mapear o esquema das fontes de dados locais em um modelo de dados comum (esquema conceitual), que é apresentado na arquitetura de integração. Na arquitetura proposta, os esquemas conceituais devem ser representados através de esquemas XML, definidos através da XML *schema*, uma linguagem de definição de esquemas para dados XML [9]. Várias propostas de mapeamento de dados armazenados em bancos de dados convencionais (por exemplo, relacionais e orientados a objeto) em documentos XML têm sido publicadas e implementadas, como por exemplo, a ferramenta XDK for PL/SQL da Oracle [21]. De qualquer forma, pode-se utilizar a seguinte estratégia genérica de mapeamento: representar os objetos (ou tabelas) como elementos XML e as propriedades (ou atributos) como sub-elementos. Essa forma de mapeamento é mais transparente e garante uma maior autonomia às fontes de dados, porém o processo de integração de dados através da linguagem de consultas é sobrecarregado com as atividades de resolução de conflitos.

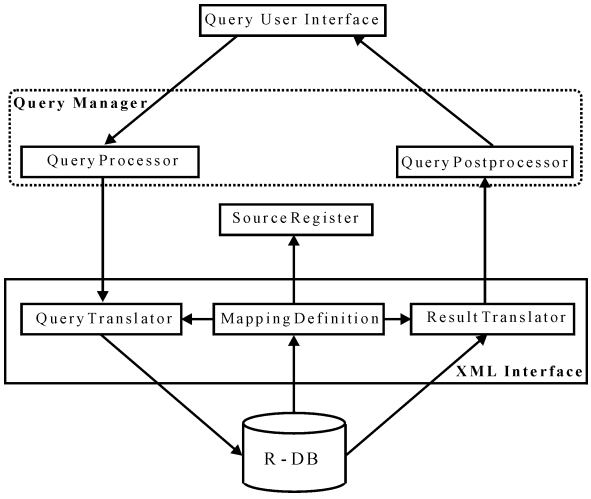


Figura 4. Detalhe da Interface XML.

A Figura 4 apresenta a interface XML com mais detalhes. A atividade de mapeamento descrita anteriormente é executada pelo componente de definição de mapeamento (*Mapping Definition*).

Além da atividade de mapeamento a interface XML deve traduzir consultas expressas no modelo de dados XML para o modelo de dados da fonte de dados local. Essa tarefa é realizada pelo tradutor de consultas (*Query Translator*). Este componente utiliza esquemas XML (definidos em XML *schema*) para efetuar o mapeamento.

O último componente da interface XML é o tradutor de resultados (*Result Translator*), que executa o papel inverso do tradutor de consultas. Esse componente deve apresentar os resultados expressos de acordo com o modelo de dados da fonte de dados local em um formato de documento XML. Por exemplo, um banco de dados relacional retorna um conjunto de tuplas para uma determinada consulta, em seguida, o tradutor de resultados deve criar elementos e sub-elementos XML que identifiquem a tabela consultada, as diversas tuplas obtidas e o conteúdo dos atributos de cada tupla. Portanto, a saída deste componente é um documento XML bem-formado.

5. Processador de Consultas

O primeiro passo executado pelo QP, ao receber uma consulta MXQuery, é checar a sintaxe e semântica da consulta recebida, onde uma árvore de expressões que simboliza a consulta original é gerada. Observando a figura 5, podem-se identificar dois componentes responsáveis por executar este primeiro passo. O primeiro faz a verificação sintática da consulta, onde uma árvore de análise obtida a partir da consulta original é comparada à gramática da linguagem MXQuery. Caso a consulta seja sintaticamente válida, o segundo componente verifica a existência dos objetos apontados na consulta no catálogo de metadados publicados. Finalmente, se todos os objetos existirem e forem declarados corretamente, é gerada a árvore de expressões de consulta global.

O otimizador global simplifica, para melhorar a performance, a consulta original avaliando diferentes planos lógicos de consulta equivalentes, que são obtidos através da aplicação de regras de transformação especificadas na álgebra que define a linguagem de consultas.

Em um banco de dados convencional, o processador de consultas deve determinar o plano físico a ser executado a partir do plano lógico selecionado durante a etapa de otimização de consultas, porém, ao integrar bancos de dados múltiplos, só se conhece a disponibilidade dos resultados parciais no momento da integração. Essa informação pode alterar a seleção do plano físico [15]. Além disso, o MDBS não possui todas as informações estatísticas necessárias a respeito das fontes de dados, conseqüentemente, é difícil optar pelo melhor plano físico antes de obter os resultados parciais.

O gerador de sub-consultas utiliza a árvore de expressões simplificada para produzir várias sub-consultas expressas em XQuery. Será produzida pelo menos uma sub-consulta XQuery para cada fonte de dados que conste na relação de fontes de dados consultadas. Como as próprias expressões XQuery podem ser simplificadas utilizando a especificação formal apresentada em [7], existe um fluxo nos 2 (dois) sentidos entre o otimizador global e o gerador de sub-consultas.

Assim, a interação entre esses dois processos determina a geração de 5 (cinco) produtos:

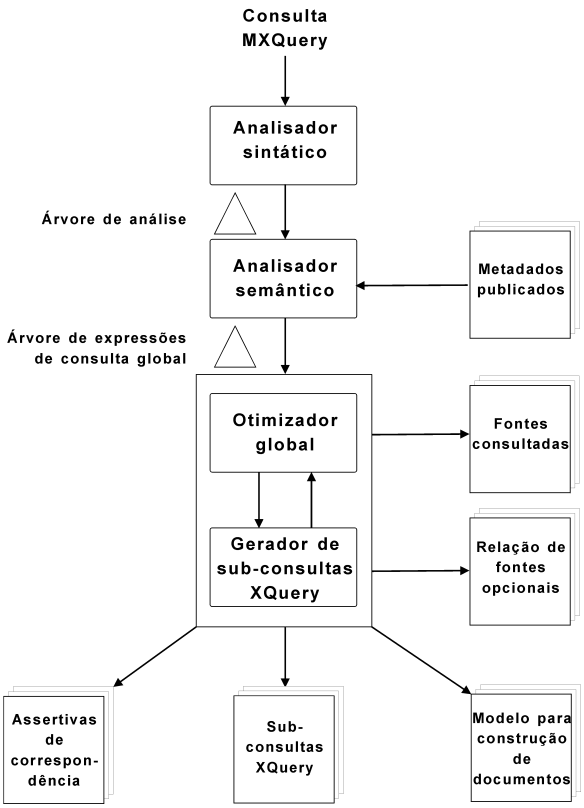


Figura 5. Produtos Gerados pelo QP.

1. Fontes consultadas. Uma lista de todas as fontes que foram referenciadas pela consulta global. Essa informação será utilizada pelo processador de localização e pelos processos de envio de consulta e recebimento de resultados;
2. Relação de fontes opcionais. A linguagem MXQuery permite a classificação das fontes de dados em obrigatórias e não obrigatórias (opcionais). Resumidamente, uma fonte de dados opcional indisponível não inviabiliza uma consulta, apenas altera parcialmente seu resultado. É produzida uma lista de todas as fontes opcionais, que será consultada quando uma fonte de dados estiver indisponível. Caso essa fonte de dados faça parte dessa relação, tal evento não interromperá a consulta;
3. Sub-consultas XQuery. É gerada pelo menos uma sub-consulta XQuery simplificada para cada fonte de dados consultada;
4. Assertivas de correspondência. É produzida uma relação com todas as assertivas de correspondência entre os elementos das fontes de dados consultadas e os elementos do documento que representará o resultado global integrado [19, 13]. As assertivas de correspondência especificam as regras para resolução de conflitos extraídas da consulta MXQuery;
5. Modelo para construção de documentos. É gerado um documento que especifica as transformações/construções que devem ser realizadas sobre os elementos originais para a produção do documento resultado.

A idéia de originar vários produtos a partir de uma consulta MXQuery é permitir a geração de sub-consultas XQuery bem simples. As transformações de elementos XML e resolução de conflitos de integração só são realizadas pela arquitetura de integração. Assim, as fontes de dados locais não necessitam realizar qualquer esforço para apresentar seus resultados. Posteriormente, o QPp utilizará os produtos originados pelo QP para construir o resultado aguardado.

Antes de enviar as sub-consultas às fontes de dados correspondentes é necessário substituir a referência lógica utilizada para identificar a fonte de dados com a localização física da referida fonte. Essa tarefa é realizada pelo processador de localização que consulta o repositório MDBS, extrai a informação necessária e substitui as referências lógicas.

Finalmente, o QP envia as sub-consultas para as interfaces XML correspondentes e aguarda o sinal de recebimento. Se decorrido um intervalo de tempo $\triangle T$ e o sinal de reconhecimento não chegar, o QP reenvia a consulta. O QP continua tentando enviar a sub-consulta durante um intervalo de tempo $\triangle T'$, porém, decorrido esse intervalo de tempo e se, ainda assim, alguma interface XML não enviar o sinal de reconhecimento, o QP executa a operação de descarte ou interrompe a consulta de acordo com os seguintes critérios:

1. Descartar. Caso a referida fonte de dados conste na lista de fontes de dados opcionais, todas as suas referências são excluídas. Assim, as regras de integração e de construção de elementos que referenciam a fonte de dados indisponível, são descartadas, bem como, a lista de fontes consultadas é atualizada. Além disso, a sub-consulta é eliminada e o processo de envio é interrompido.
2. Cancelar. Caso a fonte de dados não conste na lista de fonte de dados opcionais, o QP envia uma mensagem de erro para o QPp, envia mensagens de *"abort"* para as demais interfaces XML e descarta as tabelas temporárias (produtos do processador de consultas).

6. Conclusão

Este artigo propõe uma extensão a linguagem XQuery, denominada de MXQuery, com objetivo de utilizá-la como uma linguagem consulta (MDL) em sistemas de bancos de dados múltiplos (MDBS). A linguagem proposta permite que consultas sejam realizadas sobre fontes de dados heterogêneas. A MXQuery resolve problemas de integração de dados como heterogeneidade semântica e o tratamento de informações incompletas. A estrutura da extensão é bastante flexível, pois garante a integração de um número variável de fontes de dados com diferentes graus de autonomia.

Foi proposta ainda uma arquitetura para o processamento de consultas em sistemas de banco de dados múltiplos. A idéia é que a linguagem MXQuery seja incorporada a esta arquitetura como linguagem de banco de dados múltiplos (MDL). Atualmente, tem-se trabalhado em um protótipo da arquitetura de integração proposta neste artigo. Devido à natureza dos documentos produzidos pela arquitetura e da necessidade de manter uma independência de plataforma, tem-se utilizado a linguagem Java para implementar o processador de consultas MXQuery. Além disso, tem-se estudado a viabilidade de utilizar um banco de dados XML nativo para implementar a função do repositório MDBS. Além disso, pretende-se formalizar uma álgebra para a linguagem MXQuery a partir da álgebra proposta para XQuery. Com esta álgebra, pretende-se aproveitar um maior número de oportunidades de otimização no processamento de consultas na arquitetura proposta.

REFERÊNCIAS

[1] Abiteboul, S., Segoufin, L., Vianu, V.. Representing and Querying XML with Incomplete Information. *In Proc. of ACM PODS'01*. 2001.

[2] Boag, S., Chamberlin, D., Fernadez, M., Florescu, D., Robie, J., Siméon, J., Stefanescu, M.. XQuery 1.0: An XML Query Language. *W3C - Work in progress*. http://www.w3.org/TR/xquery/. 2003. Consultado em 12/11/2003.

[3] Bray, T., Paoli, J., Sperberg-McQueen, C., Maler, E.. Extensible Markup Language (XML) 1.0 (Second Edition). *W3C Recommendation*. http://www.w3.org/TR/REC-xml. 2000. Consultado em 19/09/2001.

[4] Brayner, A., Aguiar, J.. Sharing Mobile Databases in Dynamically Configurable Environments. *In Proc. of the 15th International Conference on Advanced Information Systems Engineering*. 2003.

[5] Chamberlin, D., Robie, J., Florescu, D.. Quilt: An XML Query Language for Heterogeneous Data Sources. *International Workshop on the Web and Databases (WebDB'2000)*. 2000.

[6] Domenig, R., Dittrich, K.. A Query Based Approach for Integrating Heterogeneous Data Sources. *In Proc of. 9th International Conference on Information Knowledge Management*. 2000.

[7] Draper, D., Fankhauser, P., Fernández, M., Malhotra, A., Rose, K., Rys, M., Siméon, J., Wadler, P.. XQuery 1.0 and XPath 2.0 Formal Semantics. http://www.w3.org/TR/xquery-semantics/. 2003. *W3C Working Draft*. Consultado em 20/02/2004.

[8] Elmagarmid, A., Bouguettaya, A., Benatallah, B.. Interconnecting Heterogeneous Information Systems. *Kluwer Academic Publishers*. 1998.

[9] Fallside, D.. XML Schema Part 0: Primer. *W3C Recommendation*. http://www.w3.org/TR/xmlschema-0/. 2001. Consultado em 21/09/2001.

[10] Grant, J., Litwin, W., Roussopoulos, N., Sellis, T.. Query Languages for Relational Multidatabases. *In Proc. of 19th VLDB Conference*. 1993.

[11] Krishnamurthy, R., Litwin, W., Kent, W.. Languages Features for Interoperability of Databases with Schematic Discrepancies. *ACM SIGMOD Record, 20 (2): 40-49*. 1991.

[12] Litwin, W., Abdellatif, A., Zeroual, A., Nicolas, B., Vigier, Ph.. MSQL: A Multidatabase Language. *Information Sciences, (49)*. 1989.
[13] Lóscio, B.. Atualização de Múltiplas Bases de Dados através de Mediadores. *Tese de Mestrado, Universidade de Federal do Ceará, Brasil*. 1998.
[14] Missier, P., Rusimkiewicz, M.. Extending a Multidatabase Manipulation Language to Resolve Schema and Data Conflicts. *In Proc. of the 6th IFIP TC-2 Working Conference on Data Semantics*. 1995.
[15] Ozcan, F., Nural, S., Koksal, P., Evrendilek, C.. Dynamic Query Optimization in Multidatabases. *Bulletin of the IEEE Computer Society Techinical Committee on Data Engineering*. 1997.
[16] Özsu, M., Valduriez, P.. Principles of Distributed Database Systems. *Prentice–Hall, 2nd edition*. 1999.
[17] Robie, J., Chamberlin, D., Florescu, D.. Quilt: An XML Query Language. *Presented at XML Europe*. 2000.
[18] Sattler, K., Conrad, S., Saake, G.. Adding Conflict Resolution Features to a Query Language for Database Federations. *In Proc. of 3rd International Workshop on Engineering Federated Information Systems*. 2000.
[19] Spaccapietra, S., Parent, C., Dupont, Y.. Model Independent Assertions for Integration of Heterogeneous Schemas. *VLDB Journal, 1(1): 81-126*. 1992.
[20] Tomasic, A., Raschid, L., Valduriez, P.. Scaling Heterogeneous Databases and the Design of Disco. *In Proc. of the International Conference on Distributed Computing Systems*. 1996.
[21] XDK, Oracle XML Developer's Kit for PL/SQL. *Oracle Technology Network*. http://otn.oracle.com/tech/xml/xdk_plsql/indesx.html. 2003. Consultado em 22/10/2003.

APÊNDICE A – Documentos Exemplo

Documento: d1.xml

```
<bib>
  <livro isbn="393">
    <titulo>Princípios de SBDs Distribuídos</titulo>
    <autor>Özsu</autor>
    <autor>Valduriez</autor>
    <ano>1999</ano>
    <preco>89.00</preco>
  </livro>
  <livro isbn= "352">
    <titulo>Implementação Sistemas BDs </titulo>
    <autor>Garcia-Mollina</autor>
    <autor>Ullman</autor>
    <autor>Widom</autor>
    <ano>2001</ano>
    <editora>Campus</ editora >
    <preco>110.00</preco>
  </livro>
</bib>
```

Documento: d3.xml

```
<lib>
  <book>
    <isbn>053</isbn>
    <title>Fundamentals of Database Systems</title>
    <author>Elmasri</author>
    <author>Navathe</author>
    <year>1994</year>
    <press>Addison-Wesley</press>
    <price>54.00</price>
  </book>
  <book>
    <isbn>013</isbn>
    <title>A First Course in DB Systems</title>
    <author>Ullman</author>
    <author>Widom</author>
    <year>1997</year>
    <press>Prentice Hall</press>
    <price>45.00</price>
  </book>
  <book>
    <isbn>352</isbn>
    <title>Database System Implementation</title>
    <author>Garcia-Mollina</author>
    <author>Ullman</author>
    <author>Widom</author>
    <year>1999</year>
    <press>Prentice Hall</press>
    <price>50.00</price>
  </book>
</lib>
```

CONTRAM: MIDDLEWARE PARA INTEROPERABILIDADE DE REDES HETEROGÊNEAS DE CONTROLADORES SEMAFÓRICOS EM SISTEMAS DE TRANSPORTES INTELIGENTES.

Lincoln Luiz de Morais
Instituto de Informática/Universidade Federal do Rio Grande do Sul
Porto Alegre, RS, Brasil, lincoln@inf.ufrgs.br

Cláudio Fernando Resin Geyer
Instituto de Informática/Universidade Federal do Rio Grande do Sul
Porto Alegre, RS, Brasil, geyer@inf.ufrgs.br

Abstract

Systems that use computational technologies in treatment the problems relative to the transit are classified as ITS or Intelligent Transportation System. Urban Traffic Systems Management manages flow of vehicles and road occupation using traffic control devices such as traffic lights and sensors and its respective controllers. Computationally, are relevance factors: the interoperability between these controllers and standardizations adopted. A urban traffic control global system (hardware and software) normally is implemented by stages, being acquired controllers of different manufacturers and models, making it difficult the integration due its proprietary technologies.

This work presents part of CONTRAM, a middleware that, treating the traffic controllers installed in road mesh based in distributed systems and computer networks paradigms, it can be used as interface between traffic management applications and control devices controllers, allowing the integration of different specifications of controllers in an only system. This work treats only the computational architecture about integration.

Keywords: ITS, middleware, CONTRAM, traffic control, SNMP, distributed systems.

Resumo

Sistemas que utilizam tecnologias computacionais no tratamento de problemas relativos ao trânsito são classificados como *ITS* ou *Intelligent Transportation System.* Sistemas de Gerenciamento de Tráfego Urbano gerenciam o fluxo de veículos e a ocupação da malha viária utilizando dispositivos de controle de tráfego como semáforos e sensores e seus respectivos controladores. Computacionalmente, são fatores relevantes: a interoperabilidade entre estes controladores e padronizações adotadas. Um sistema global (*hardware* e *software*) de controle de tráfego urbano normalmente é implementada por etapas, sendo adquiridos controladores de diferentes fabricantes e modelos, dificultando a integração entre os mesmos em função de suas tecnologias proprietárias.

Este trabalho apresenta parte do *CONTRAM*, um *middleware* que, tratando os controladores de tráfego instalados ao longo da malha viária baseado nos paradigmas de sistemas distribuídos e redes de computadores, possa ser utilizado como interface entre as aplicações computacionais de gerenciamento de tráfego e os controladores de dispositivos de controle, permitindo a integração de diferentes especificações de controladores em um único sistema. Os aspectos tratados neste trabalho dizem respeito apenas à integração do *CONTRAM* com os controladores de dispositivos de controle.

Palavras chaves: *ITS*, *Intelligent Transportation Systems*, *Middleware*, *CONTRAM*, Controle de Tráfego Urbano, *SNMP*, Sistemas Distribuídos.

1. Introdução

A Engenharia de Tráfego representa o ramo da Engenharia de Transportes que se ocupa do planejamento, projeto geométrico, operações de tráfego em redes viárias e terminais, assim como o relacionamento com outros modos de transporte [5]. Associado à falta de planejamento urbano efetivo no desenvolvimento industrial e/ou comercial e, conseqüentemente, na expansão territorial e populacional de importantes cidades ao redor do mundo, surgiram vários problemas de impacto social, dentre eles, o da ocupação da malha viária e os problemas específicos decorrentes dessa ocupação, tais como congestionamentos, atropelamentos e assaltos no trânsito, com ou sem vítimas. Obras civis como construções de viadutos, de novas vias ou alargamento das já existentes ou ainda melhorias no sistema de transportes coletivos, podem contribuir positivamente para minimizar os problemas acima citados. Outras alternativas, a um custo inferior, surgem a partir ou da criatividade humana, como é o caso do sistema de rodízio adotado pela cidade de São Paulo, ou a partir da utilização de tecnologia computacional no controle e gerenciamento do tráfego urbano, no qual se incluem os sistemas que buscam minimizar congestionamentos urbanos e suas conseqüências, originando termos como *CATE* (*Computer Aided Traffic Engineering)* e *ITS*.

Buscando atingir os objetivos propostos, os SGTU's ou Sistemas de Gerenciamento de Tráfego Urbano atuam coordenando e sincronizando os tempos semafóricos de um grupo de semáforos, através de seus controladores, em função das necessidades da demanda, detectadas através de dados obtidos a partir de outro tipo de dispositivo de controle, os sensores, localizados ao longo das vias e atuando como monitores das condições do tráfego.

A utilização de tecnologia inteligente, baseada em microprocessadores, nos controladores atuais permite um escopo maior de funções dentro da atividade de controle e monitoramento, porém, muitas destas facilidades não são suportadas pelos sistemas mais antigos, ou até mesmo de uma geração anterior, devido a falta de um padrão aberto que possibilite a interoperabilidade, já que cada fabricante adota soluções proprietárias. Para a captura automática, e realisticamente mensurados, dos dados de entrada sobre os quais o SGTU realiza operações para a tomada de decisão, faz-se necessário a integração deste com os controladores e entre os próprios controladores, valendo-se de todas as facilidades tecnológicas na obtenção dos dados. Uma ferramenta de controle de tráfego urbano normalmente é implementada por etapas, em que diferentes especificações de controladores são adquiridos por motivos diversos. Tais diferenças freqüentemente geram incompatibilidades nas trocas de dados. Portanto é interessante buscar uma forma de viabilizar a comunicação entre todos esses equipamentos, de forma que investimentos realizados possam ser preservados através do estabelecimento de um formato padrão para a troca de dados entre os mesmos, permitindo o rompimento com a dependência atual causada pelo uso de sistemas proprietários sem contudo desprezar o conhecimento acumulado. A principal crítica por parte dos usuários tem sido a inflexibilidade dos sistemas disponíveis comercialmente, muitos são tidos como "caixas-pretas" [5].

Este trabalho buscou nos paradigmas de sistemas distribuídos e de redes de computadores, uma alternativa ao problema da interoperabilidade, apresentando um modelo de rede de controladores voltado ao apoio à gestão e operacionalização do tráfego urbano, que possa ser utilizado como interface entre um SGTU e diferentes modelos e tipos de controladores em um único sistema, permitindo que controladores com ou sem inovações tecnológicas possam ser adicionados e/ou removidos, de forma transparente à aplicação de gerenciamento, ou seja, sem que esta sofra alterações em seu código fonte, promovendo sua revitalização e adequação de acordo com necessidades específicas.

O escopo do modelo é prover mecanismos, mapeamento de recursos, comunicação e conversão de dados, que permitam atender às operações, consulta e configuração dos valores das variáveis utilizadas para controlar, monitorar e gerenciar o fluxo de veículos, solicitadas pelo SGTU, liberando-o de conhecer detalhes técnicos tanto dos controladores como dos envolvidos na conversão e troca de dados. Está fora do escopo interpretar as operações solicitadas pelo SGTU. O modelo foi batizado de ***CONTRAM***, um acrônimo de ***CON****trollers* ***TRA****ffic* ***M****iddleware*. Neste trabalho será apresentado apenas o modelo da solução adotada para integrar os controladores ao *CONTRAM*, embora o modelo originalmente proposto baseie-se em uma arquitetura multicamadas, 4-*tier*, podendo atuar em uma configuração centralizada ou distribuída, dependente das características do SGTU, trocando dados através da Internet, e projetado tendo em mente a utilização de padrões abertos da indústria da Informática e de Transportes, a interoperabilidade entre diferentes elementos tecnológicos de controle de tráfego e a expansibilidade de um SGTU.

Este trabalho é composto por 4 seções. A próxima seção trata de conceitos básicos do tráfego urbano e seu controle; em seguida são apresentadas as características do *CONTRAM* para a integração dos controladores e por fim as conclusões.

2. Tráfego urbano e seu controle

Genericamente, um sistema de tráfego é composto por vias, interseções e pelo movimento de elementos circulantes, veículos e pedestres, que são os usuários desse sistema. O movimento dos veículos originam as chamadas correntes de tráfego ou fluxo.

Dispositivos de controle como os semáforos buscam otimizar o fluxo através do sincronismo dos tempos semafóricos ou tempo de duração dos intervalos de verde e vermelho. O conjunto de tempos semafóricos é chamado de *plano semafórico*, que pode ser concebido dinamicamente em função da demanda momentânea ou pré-programado com base em dados históricos do comportamento do tráfego em dias e horários específicos. Estes planos são ativados em função da variação do comportamento do fluxo, em diferentes horários do dia e dias da semana segundo uma tabela de horários de ativação, ou ainda em condições específicas, como por exemplo, uma manifestação pública e detecção de falha no equipamento.

A Figura 2.1 exemplifica uma interseção ou cruzamento entre uma avenida de via dupla e uma rua de via simples. Por grupo entende-se um conjunto de semáforos que possuem a mesma fase ou intervalo em determinado instante. No exemplo são tratados dois grupos semafóricos para veículos, indicados por G1 e G2, e três grupos para os pedestres, indicados por P1, P2 e P3. A Figura 2.2 mostra um exemplo de um possível plano semafórico simplificado para esta interseção.

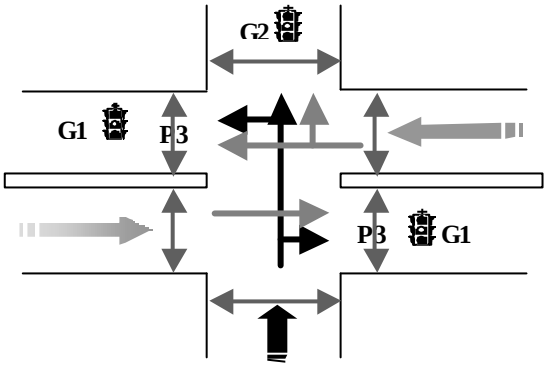


FIGURA 2.1 - Exemplo de uma interseção, com indicadores de sentido das vias e movimentos, para pedestres e veículos, e grupos semafóricos para veículos (G1 e G2) e pedestres (P1, P2 e P3).

		Seqüência em que as lâmpadas acendem (evento paralelo para diferentes fases (F1 à F5))							
Grupo	**Fase**	1	2	3	4	5	6	7	8
G1	F1	V	V	A	R	R	R	R	R
G2	F2	R	R	R	R	V	A	R	R
P1	F3	V	r	R	R	R	R	V	V
P2	F4	R	R	R	R	V	V	V	r
P3	F5	R	R	R	R	R	R	V	r
		16	1	1	1	8	1	3	3
		Tempo em que as lâmpadas permanecem acessas ao longo da Fase.							

FIGURA 2.2 – Exemplo de plano semafórico simplificado para dada interseção, onde o ciclo se inicia em verde (*V*), passando pelo amarelo (*A*), vermelho (*R*) e vermelho de pedestre (*r*).

Os semáforos funcionam conectados e recebendo dados de um equipamento eletrônico microprocessado conhecido por controlador de dispositivos de controle, ou simplesmente, controlador. Este equipamento possui recursos voltados à comunicação de dados, à realização de um pequeno volume de processamento e ao armazenamento dos planos semafóricos e tabelas diversas, estando todo o seu funcionamento baseado nos planos semafóricos, cujos tempos das fases dimensionados nos mesmos são os principais parâmetros de entrada ao processamento a ser executado. Estes planos semafóricos podem ser programados através de um equipamento específico conhecido como *Programador de Controlador* ou através da aplicação que gerencia o tráfego, permanecendo armazenados na memória *RAM* do controlador quando estiver habilitado ou for o plano semafórico vigente, e em memória do tipo *EPROM*, também no controlador, quando estiver desabilitado. Outra alternativa é armazenar estes planos em um microcomputador que executa o SGTU e está conectado ao controlador a ser atuado.

Segundo [8] e [3], os semáforos podem ser classificados em função do seu modo de operação ou estratégia operacional, que é definido pelo controlador ao qual o mesmo está conectado. Um destes modos operacionais é conhecido como *modo central*, ou seja, quando o plano semafórico a ser ativado é escolhido pela central de controle através de um SGTU.

Quando um controlador está conectado a uma rede, diz-se que o mesmo está operando na modalidade *Coordenado* [8], ou seja, está obrigado a respeitar uma defasagem de tempo no início de cada ciclo, garantindo, por exemplo, a chamada *onda-verde*. Uma rede poderá conter um número máximo de controladores, sendo um deles designado para gerar o sincronismo da rede. Este é o gerente de comunicação ou referencial, com capacidade de consultar e alterar a programação de qualquer outro controlador sob sua coordenação. O comando de *forçamento remoto* faz com que qualquer controlador da rede, ou todos, funcionem de maneira diferente do programado no plano semafórico durante um período de tempo, permitindo que a rede se comporte de acordo com uma necessidade momentânea do controlador referencial. Fisicamente, os controladores são interligados por uma interface serial, numa configuração multiponto, através de cabos metálicos, permitindo comunicações confiáveis até uma distância máxima de 3500 mts [3] e 5000 mts [8].

3. *CONTRAM*

[4] define como arquitetura de sistema um modelo que expressa as funcionalidades do sistema e a integração entre estas funcionalidades em um alto nível de abstração, podendo existir diversas sub-arquiteturas ou camadas para que o mesmo seja abrangente, hoje e no futuro. As finalidades básicas da elaboração de uma arquitetura de sistema são: a) Construir sistemas integrados eficientemente; b) Garantir a expansibilidade do sistema; c) Promover padrões. Dada a abrangência dos sistemas para *ITS* é de fundamental importância a elaboração de uma arquitetura de sistema, por envolver diversas áreas sociais que são suscetíveis a mudanças comportamentais.

3.1 Arquitetura funcional do *CONTRAM*

A arquitetura do *CONTRAM* foi estruturada em sub-arquiteturas, que por suas vez foram subdivididas em camadas onde estão definidas as funcionalidades do mesmo. O objetivo da especificação e implementação de uma arquitetura multicamadas é garantir a coesão, a modularidade e a expansibilidade do sistema como um todo, uma vez que para agregar novas funcionalidades ou rever as já existentes, modifica-se a camada que fornece essa funcionalidade, seja ela uma nova tecnologia ou um novo serviço, permitindo, do ponto de vista comercial, a criação de novas oportunidades de negócios e mercados. Basicamente as camadas apresentadas pelo modelo são: a) *Apresentação*; b) *Regras de Negócios*; c) *Infra-estrutura*; d) *Dados*, integradas conforme Figura 3.1

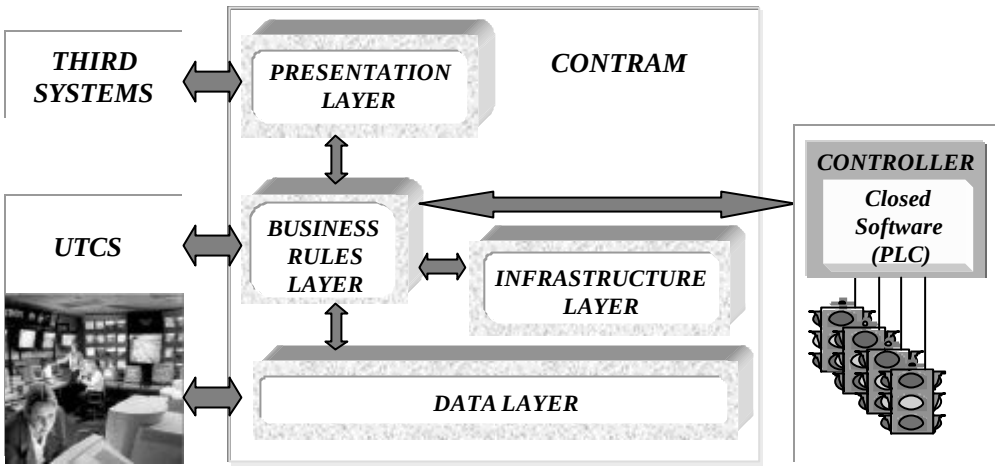


FIGURA 3.1 – Integração entre as múltiplas camadas do *CONTRAM*, onde as setas indicam as interações entre as camadas e sistemas (SGTU e Controladores).

Tipicamente, os dados em uma aplicação de *ITS* são gerados por diferentes tipos de sensores geograficamente distribuídos ao longo de uma área de controle de tráfego, também chamada de CTA, que mensuram diferentes parâmetros da aplicação, cada um fornecendo um panorama da situação momentânea. A infra-estrutura de comunicação e tecnologias de dispositivos de controle de tráfego estão permitindo projetos de aplicações que funcionem em ambiente de computação distribuída, combinando os dados obtidos para oferecer diversos serviços aos clientes, também distribuídos [1]. O modelo *CONTRAM* assume a cidade dividida em CTA's, cada qual com uma rede de controladores e gerenciada por um computador regional ou *Inspector Computer* (*IC*), que interage com o computador do centro operacional ou *Delegate Computer* (*DC*) onde está instalado o SGTU.

Com relação ao empacotamento de *software*, apenas conceitualmente, foram projetados dois módulos logicamente equivalentes, o *Central Module* (*CM*) que é instalado no *Delegate Computer* atuando como interface com o SGTU e o *Remote Module* (*RM*) instalado nos computadores que estão distribuídos ao longo da malha viária, ou *Inspector Computer*, atuando como interface com os controladores. Os serviços previstos podem ser ativados ou não em função do local de execução. Ambos os módulos acessam bases de dados que armazenam os dados manipulados pelo *CONTRAM*. Esses dados podem conter valores resultantes de processamento realizado ou parametrizações utilizadas pelo modelo. A visão global dos dados do sistema é mantida no *DC* e as visões parciais, referentes aos controladores das CTA's administradas, são mantidas no *IC*. O domínio do *CONTRAM* limita-se ao atendimento das solicitações de operações de atuação sobre os controladores solicitadas pelo SGTU, não se responsabilizando por quaisquer decisões tomadas a nível de gerenciamento. Atendidas as normas pré-definidas pelo modelo para a integração, toda e qualquer solicitação de operação de consulta ou configuração recebida de um sistema de controle será atendida, não sendo avaliado o seu impacto no comportamento do tráfego urbano. A Figura 3.2, contextualiza o ambiente controle de tráfego como um todo, demonstrando a arquitetura funcional do *CONTRAM* em alto nível e a delimitação do seu escopo .

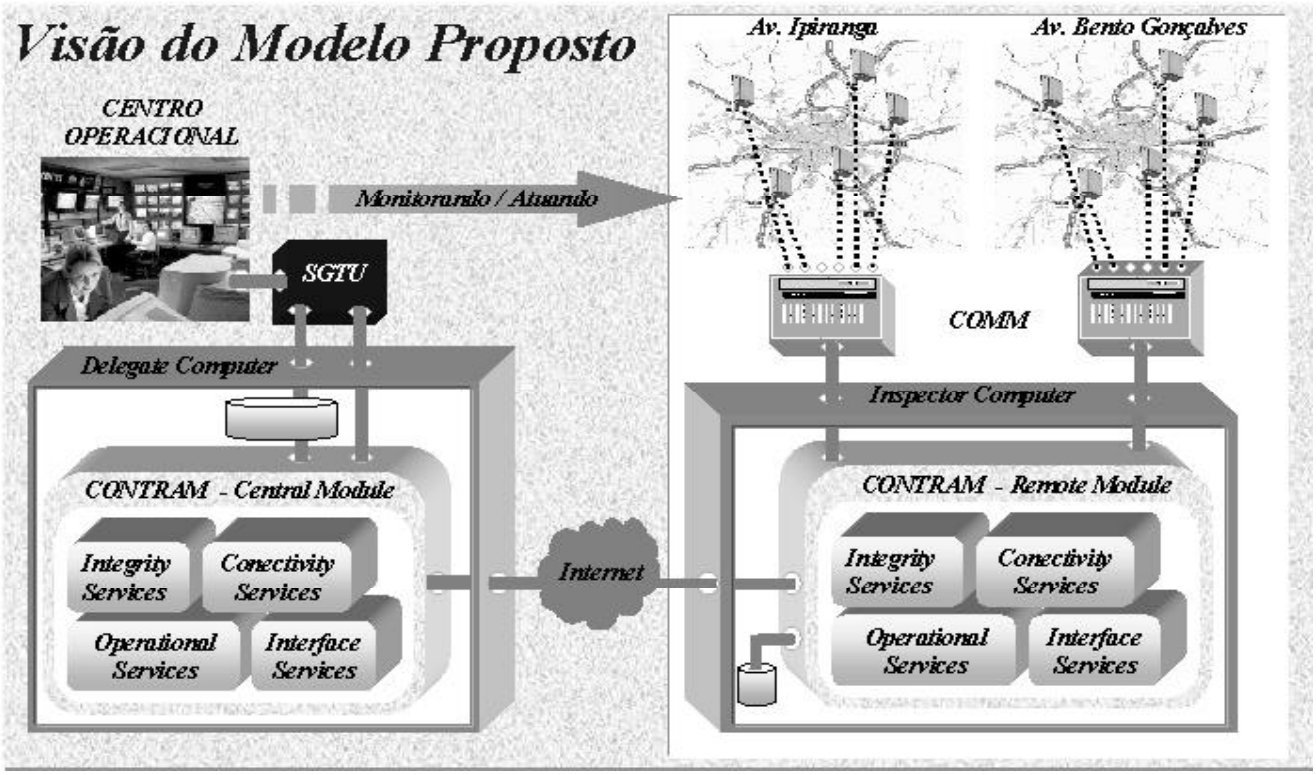


FIGURA 3.2 – Contexto de controle de tráfego, demonstrando a arquitetura funcional e a delimitação do escopo do *CONTRAM*.

Dado o objetivo do *CONTRAM*, são definidas duas interfaces com características distintas de concepção e funcionamento, uma com os sistemas de alto-nível ou os SGTU's e outra com os sistemas de baixo-nível ou os controladores de dispositivos de controle de tráfego, sendo o escopo deste trabalho limitado a esta última.

3.2 Integração *CONTRAM* - Controladores

As funcionalidades providas pelo modelo são chamado de serviços, que englobam os subserviços. Dentre os serviços providos pelo modelo existem aqueles que são chamados de *nativos*, constituindo-se na base de funcionamento do mesmo e os *agregados*, que dão funcionalidades extras podendo ser alterados de forma que o funcionamento básico não seja alterado. Esses serviços foram agrupados hierarquicamente baseados em sua abrangência funcional dentro do funcionamento do *CONTRAM*, conforme Figura 3.3.

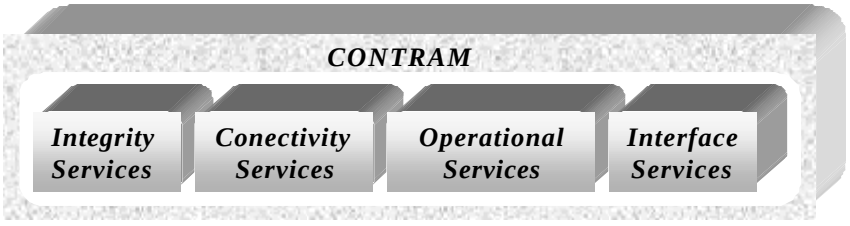


FIGURA 3.3 – Hierarquia dos serviços previstos pelo *CONTRAM*.

Para a integração do *CONTRAM* com os controladores são utilizados subserviços de *Interface Services* e *Conectivity Services*, voltados respectivamente ao interfaceamento com os sistemas computacionais a serem integrados e à transferência de dados. Os serviços de conectividade são utilizados em duas situações distintas, onde uma delas é para permitir a troca de mensagens entre o *CONTRAM*, através de um módulo de *software* com mecanismos de gerente, e os controladores, através de módulos de *software* com mecanismos de agente que convertem as mensagens recebidas do módulo gerente para serem processadas. Nesse caso é utilizado o protocolo *SNMP*, coexistindo as figuras do gerente e agente *SNMP* [2], [6]. De forma a preservar a arquitetura de *hardware* dos controladores, optou-se por trabalhar com agentes do tipo *proxy* e instalados no *IC* ao qual o controlador referencial está conectado.

A ferramenta que está sendo utilizada para o desenvolvimento dos serviços de integração entre o *CONTRAM* e os controladores, o *Java Dynamic Management Kit* ou *JDMK* [7], oferece algumas funcionalidades adicionais que foram incorporadas ao modelo proposto. Dentre elas está o conceito de *cascading* ou hierarquia de agentes, onde há um *Master Agent* que distribui os serviços de supervisão para um grupo de agentes subalternos, os *Leaf-node Agents*, gerenciando-os. Esses *Leaf-node Agents* são componentes administráveis chamados de *Mbeans* e responsáveis pela interação com os recursos a serem gerenciados, neste caso os controladores. Esses *Mbeans* são os próprios agentes-*proxy*, podendo ser integrados, retirados ou atualizados em tempo real, sem a modificação de outros componentes do sistema como um todo. Um *Master Agent* suporta vários agentes-*proxy*. A Figura 3.4 ilustra o funcionamento da tecnologia.

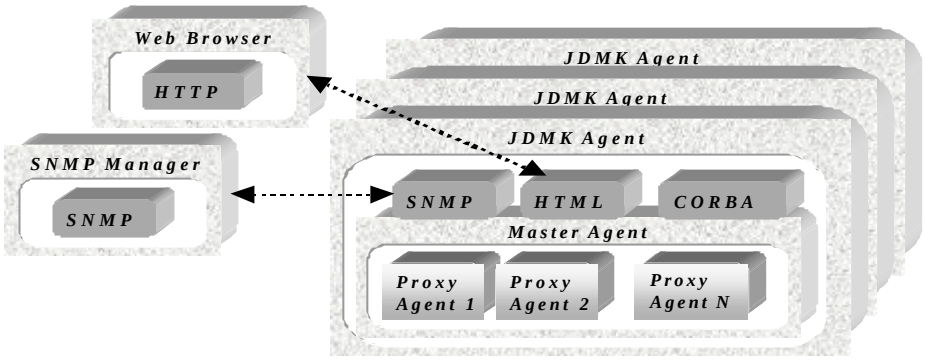


FIGURA 3.4 – Estrutura hierarquica de agentes dinâmicos, que define um "ambiente" agente (*JDMK Agent*) que contém adaptadores de protocolos, um agente principal (*Master Agent*) e agentes secundários (*Proxy Agent*).

Cada controlador possui o seu respectivo agente-*proxy* que permanece ativo o tempo todo trocando mensagens *SNMP* com um módulo gerente, nativo do *CONTRAM*. Esse agente-*proxy* atua como uma *API* do controlador, conforme Figura 3.5, à frente.

Considerando as possíveis formas de distribuição de recursos permitidos pelo *CONTRAM*, que os agentes-*proxy* serão referenciados por um mecanismo nativo do modelo, o gerente *SNMP*, e coexistem vários em um mesmo *IC*, foi adotada uma padronização quanto nomenclatura e localização dos mesmos, ambas definidas no instante de sua instalação, visando organização e controle sobre o ambiente operacional. O nome do agente-*proxy* deve ser o mesmo utilizado para identificar o controlador ao qual irá converter as mensagens, que é único em todo o sistema, e a sua localização deve estar associada a um repositório dentro da estrutura de repositórios criada pelo usuário administrador, juntamente como os demais dados por ele manipulados. O *CONTRAM* classifica um controlador como um usuário direto, sendo necessário o seu cadastramento e de seus parâmetros, permitindo sua identificação, localização e efetiva integração ao sistema como um todo. Integração esta que dá-se através do método *plug in* com ligação dinâmica, de maneira tal que um agente-*proxy* pode evoluir em suas funcionalidades refletindo a própria evolução do controlador. Quando se fizer necessário a sua atualização, deve-se desativa-lo e sobrepor o seu

conteúdo mantendo-se os mesmos parâmetros relativos à localização e identificação, de forma que quando o mesmo for ativado, o *Master Agent* e o gerente *SNMP* passam a interagir com esse novo agente-*proxy* suportando as novas funcionalidades, conforme Figura 3.4. Com isso mantém-se a modularidade do sistema como um todo.

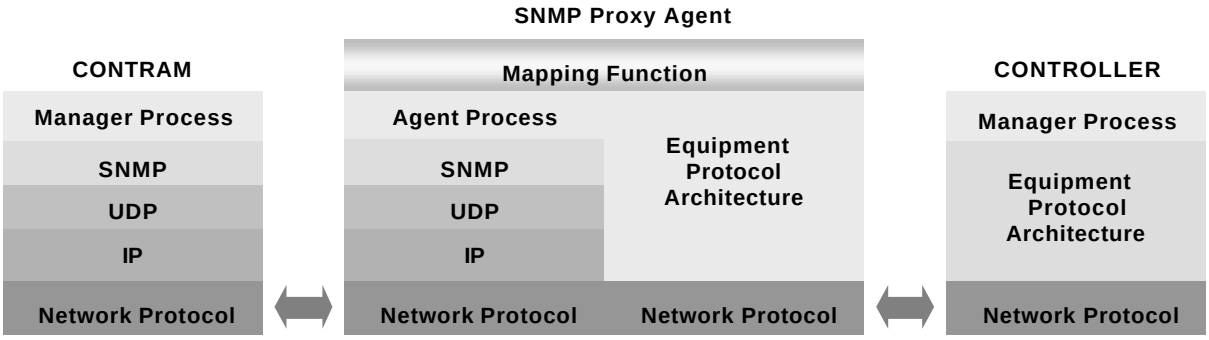


FIGURA 3.5 – Estrutura interna de um agente-*proxy SNMP*.

Quanto à concepção desse agente-*proxy*, de inteira responsabilidade do fornecedor do controlador, sugere-se que seja possível configurar parâmetros com relação a identificação e localização de recursos com o qual o mesmo interage, como as *MIB*'s, os controladores e o gerente *SNMP*, permitindo eventuais reconfigurações do ambiente, devido à critérios organizacionais ou em ocorrência de falhas.

As Figuras 3.6 e 3.7 demostram, respectivamente, o diagrama de seqüência simplificado para uma operação de obtenção do valor de um dado do controlador e o diagrama conceitual da troca de dados na interface específica.

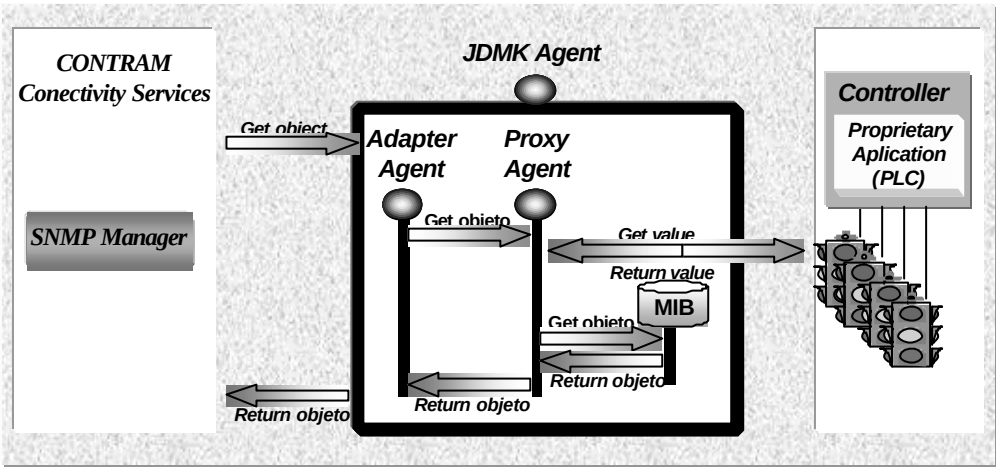


FIGURA 3.6 – Diagrama de seqüência de operações de obtenção de um valor junto ao controlador.

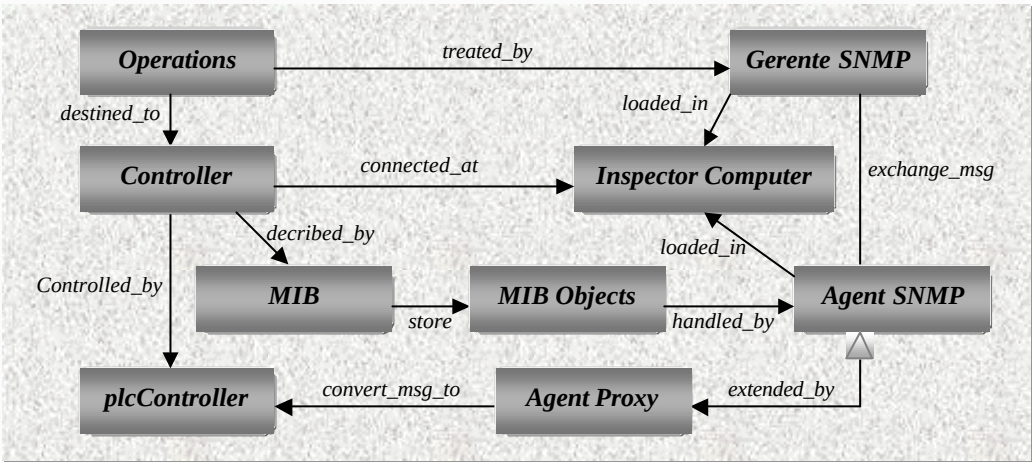


FIGURA 3.7 – Diagrama conceitual de uma troca de dados envolvendo *CONTRAM* e os controladores.

3.3 Organização dos recursos gerenciáveis nos *IC*

A implementação do agente para atuar sobre o controlador varia de acordo com o modelo e arquitetura deste, sendo identificadas algumas alternativas para tal. São elas: *a)* inserir um agente *SNMP* no módulo de controle do controlador. Considerando que o mesmo já possui suporte aos protocolos da pilha *TCP/IP*, o acréscimo de um agente *SNMP* e sua *MIB* poderia causar algum impacto no ciclo de execução das instruções. Entretanto, no caso de não apresentar suporte ao *TCP/IP*, o custo de projeto e implementação seria, provavelmente, alto. Outro agravante seria em relação a uma provável alteração na sua arquitetura de *hardware*, considerando a inclusão do agente, *MIB* e a pilha de protocolos; *b)* implementar o agente *SNMP* no módulo de comunicação. Essa alternativa minimiza o impacto na arquitetura do sistema como um todo, já que somente um módulo auxiliar do controlador foi alterado. Nesse caso, o módulo de comunicação deve dar suporte, além da pilha de protocolo *TCP/IP*, ao agente-*proxy*, já que o *SNMP* não é nativo; *c)* outra alternativa, que foi adotada pelo *CONTRAM*, é implementar o agente em um equipamento de uso genérico, como é um microcomputador, permitindo um escopo maior na função supervisória. Uma das vantagens dessa alternativa é que fica preservada a arquitetura do controlador, estando a sua *MIB* e o seu agente-*proxy* instalado no microcomputador ao qual o mesmo esta conectado. O benefício direto dessa alternativa é a menor relação custo x benefício para a integração do controlador com o mundo *SNMP*, colocando toda a dificuldade dessa integração sob a responsabilidade do *CONTRAM*. A Figura 3.8 ilustra essa alternativa de implementação.

FIGURA 3.8 – Agente *SNMP proxy* localizado em *IC* ao qual o controlador está conectado.

Cada controlador referencial possui um único agente-*proxy*, instado no *IC* ao qual o controlador está conectado, sendo utilizando o agente gerente *SNMP* configurado no *remote model* para trocar dados com o SGTU.

O fato do *CONTRAM* trabalhar com localização de recursos parametrizada permite uma grande flexibilidade quanto à distribuição dos dados. Um aspecto relativamente importante que pode causar problemas de inconsistências, dada essa liberdade de nomenclatura e localização, é a falta de critérios, padronizações e gerenciamento na escolha destes parâmetros, como nomes e endereços. Logo é importante observar critérios de nomenclatura e localizações.

Este trabalho sugere uma padronização de nomenclatura e localização para as *MIB*'s, base de dados, controladores, agentes-*proxy* e agentes-*adapter*, que são os recursos gerenciáveis com um maior número de ocorrências no modelo, visando organização e controle sobre o ambiente operacional, agregando aos mesmos algumas informações que facilitam a sua identificação em função do tipo, conteúdo e localização lógica e física de instalação. Deve ser salientado que o administrador tem a liberdade de atribuir nomes e criar uma estrutura organizacional da forma que lhe for mais conveniente dentro do seu modelo de malha viária ou distribuição física dos recursos; tanto nome como localização devem ser inseridos na arquitetura do *CONTRAM* no instante da instalação do recurso.

Quanto à nomenclatura, é sugerido um prefixo identificador e um sufixo quantificador. O prefixo é indicado para identificar o tipo de recurso, conforme Tabela 1, e o sufixo para diferenciar um único recurso dentre os vários similares, devendo ser um algarismo alfanumérico com duas casas, com valores variando entre 00 e ZZ e indicando uma ordem de precedência. Tanto o nome do *DC* ou *IC* é definido pelo usuário administrador. Para compor os nomes das bases de dados são utilizados o nome do computador, *inspector ou delegate*, no qual as mesmas estão localizadas associando-se o prefixo referente ao tipo de parâmetros que os dados armazenados se referem, conforme Tabela 1.

TABELA 1 – Prefixos identificadores dos tipos de recursos gerenciáveis.

Tipo de Recurso	***Prefixo***
Delegate Computer	*DC*
Inspector Computer	*IC*
Controlador	CTRL
Base de dados	*DBRO, DBNR, DBDR, DBHO, DBPO ou DBMR*
Agente *proxy*	APX
Agente *adapter*	AAD
MIB	*MIB*

Quanto ao controlador, há a possibilidade de existir mais de um sendo gerenciado por um mesmo *IC*, tal que sua nomenclatura deve herdar o nome do *IC* ao qual está conectado mais um prefixo determinado e correspondente sufixo. Quanto ao nome da *MIB*, deve-se identificar a qual controlador pertencem os objetos nela contidos, sendo indicado o nome do controlador com específico prefixo; os agentes-*proxy* e agentes-*adapter* devem indicar para qual controlador estão sendo convertidas as mensagens ou sobre qual *MIB* está atuando, também com respectivo prefixo, conforme Tabela 1.

Dada a coexistência de diversos recursos gerenciáveis em um único *IC*, a sugestão de organização se estende à localização dos mesmos. Cada conjunto individual de recursos pertinentes a um controlador deve estar em um repositório, cujo seu nome é semelhante ao do próprio controlador, conforme Figura 3.9, e os recursos compartilhados, como as bases de dados, devem ficar em um repositório identificado pelo nome do *IC*. Toda a hierarquia de repositórios deve estar definidas dentro de um repositório principal, denominado CONTRAM, conforme figura 3.10.

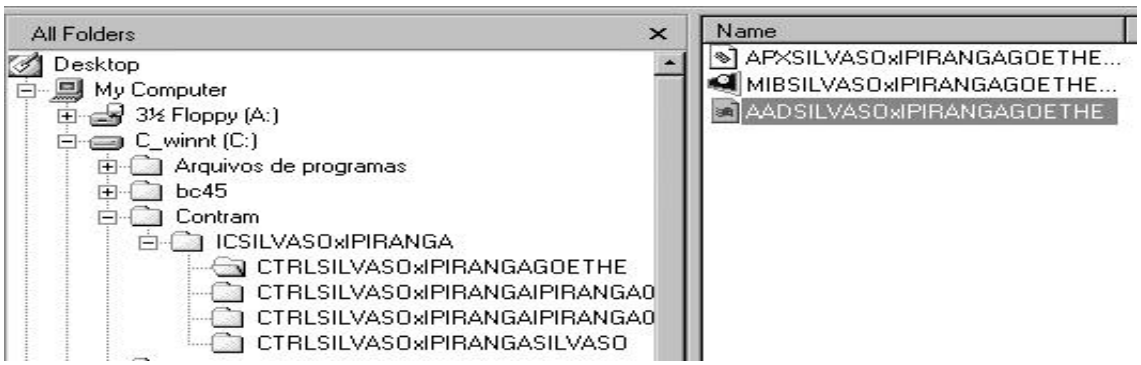


FIGURA 3.9 – Hierarquia de repositórios para um controlador e seus recursos.

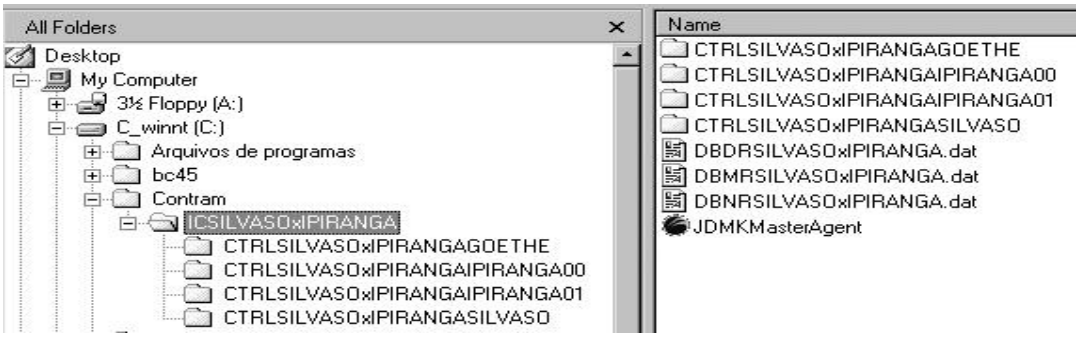


FIGURA 3.10 – Hierarquia de repositórios para um *IC* e seus controladores.

4. CONCLUSÕES

O CONTRAM atinge o seu objetivo de integrar diferentes controladores a um único SGTU. A seguir uma análise mais detalhada nos quesitos modularidade, longevidade e facilidade de integração.

Modularidade: com relação aos controladores, os fabricantes devem fornecer o agente-*proxy* correspondente. Uma vez esse agente-*proxy* ativado e o controlador já mapeado pelo *CONTRAM*, o mesmo está apto a receber dados. Caso haja troca de modelo de controlador, basta atualizar o agente-*proxy*. O *CONTRAM* interage com controlador referencial e este com os demais controladores da sua rede permitindo a integração entre redes de controladores diferentes. A limitação do *CONTRAM* está no fato de não interagir individualmente com os controladores. Para tanto seria necessário modificações na estrutura do *JDMK Agent*, conforme Figura 3.4, que deveria agregar mecanismos de agente gerente e não somente agente. Características da ferramenta *JDMK* permitem a criação de agentes com esse perfil com relativa facilidade, porém essa modificação no *JDMK Agent* agrega custos à solução.

Longevidade: uma grande e importante vantagem do *CONTRAM* é que toda a padronização proposta está explícita ao mesmo. Está armazenada em forma de *MIB* permitindo que características das inovações tecnológicas nos dispositivos de controle possam ser inseridas na mesma, sem que seja necessário modificações no *core* do CONTRAM. Qualquer dispositivo de controle que definir uma *MIB*, e fornecer um agente-*proxy* está apto a ser integrado ao CONTRAM, independentemente do seu avanço tecnológico. A sua limitação atual é integrar equipamentos de *hardware* que possuam dados complexos, como imagem ou grandes tabelas por exemplo, que não possam ser mapeados para objetos *MIB.*

Facilidade de integração: na interface com os controladores o *CONTRAM* utilizou uma tecnologia padronizada, estável, consolidada e largamente utilizas na indústria da Informática, minimizando as dependências tecnológicas, que foi o protocolo *SNMP* e objetos dispostos em formato *MIB*. Associado ao modelo de desenvolvimento de agentes, conforme Figura 3.4 e localização do agente, conforme Figura 3.8, não necessita-se de nenhuma modificação na arquitetura do *hardware* dos controladores para que o mesmo seja integrado. Os fabricantes devem utilizar um *software* compilador de *MIB* para gerar automaticamente como saída um agente a partir de uma entrada comum que é a *MIB* padronizada, tal que em um primeiro instante todos os agentes gerados serão iguais. O que difere um dos demais é justamente as diferentes características de cada controlador que serão customizadas através de linhas de código inseridas neste agente genérico que foi gerado. Por exemplo, uma empresa que possui cinco diferentes modelos de controladores, deverá possuir cinco diferentes agentes-*proxy*, um para cada modêlo, mas que foram gerados a partir de um agente genérico. A única exigência que se faz é que os agentes devam estar escritos na linguagem "C" ou Java.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Dailey, H.P. and Meyers, D. **A Structured Approach to Developing Distributed Network Applications for ITS Deployment**, Departament of Electrical Engineering, University of Washigton and Federal Highway Administration, Seattle, WA, 1997.

[2] David P. and McGinnis E. **Understanding SNMP MIBs**, Prentice Hall Inc, 1997, New Jersey.

[3] Digicon, I.C. **DIGICOM CDC 100 Manual de Operação**. Porto Alegre, dez. 1993. Relatório técnico.

[4] **System Architecture for ITS in Japan, Draft por Japan Government bodies**, Novembro/99, disponível por WWW em http://www.itsa.org/committe.nsf, em 02/2000.

[5] Rossetti, R.J.F. **Um Ambiente para Suporte à Simulação de Sistemas de Tráfego Urbano**, dissertação de mestrado, CPGCC, UFRGS, 1998.

[6] Stallings, W. **SNMP, SNMPv2, and RMON – Pratical Network Management**, Second Edition, Addison-Wesley Longman Inc., 1996.

[7] SUN MICROSYSTEMS, Inc. **Java Dynamic Management Kit White Paper**. Palo Alto, CA, 2000.

[8] Tesc, I.C. **Flexcon III, Manual de Operação**, julho de 1994.

Interactive Construction of Classification Trees Using Treemaps

Manoel Gomes de Mendonça Neto
Universidade Salvador, Nuperc,
Salvador, Brasil, 40171-100
mgmn@unifacs.br

and

Daniela Soares Cruzes
Universidade Salvador, Nuperc,
Salvador, Brasil, 40171-100
daniela@unifacs.br

and

Christiane da Costa Santana
Universidade Salvador, Nuperc,
Salvador, Brasil, 40171-100
christiane.santana@unifacs.br

Abstract

Most of the approaches published in the literature proposes a completely automatic process to generate decision trees. These approaches miss valuable expert tacit knowledge input during the construction of the tree. This paper describes an approach for interactive construction of decision trees. The approach is user-centered. It combines the strengths of the user and the computer to build better decision trees. The user provides domain knowledge and evaluates intermediate results of the algorithm. The computer automatically creates patterns satisfying user constraints and generates appropriate visualizations of the produced tree. A tool was developed to support this approach. It combines treemap visualization, visual data mining mechanisms, and the J48 (Weka) algorithm to interactively build a decision tree.
Keywords: Visual Data Mining, Decision Tree, Classification.

1. INTRODUCTION

During the last decade, data repositories has grown faster than our ability to analyze them. The area of Knowledge Discovery from Databases (KDD) has appeared as an attempt to balance this equation. KDD aims to extract non trivial, previously unknown and potentially useful information from data repositories [9],[15]. KDD is a highly interactive process that goes from the definition of the analysis goals to the extraction and assimilation of knowledge from the data repository.

The main activity of this process is data mining. Data mining is characterized by the use of algorithms to extract useful knowledge from pre-processed data sets. Classification is one of the most common tasks in data mining [15], and the construction of decision trees is one of the most popular classification methods. Decision Trees are intuitive, easy to interpret; relatively fast to construct and, compared to other classification methods, have equal or better accuracy [8].

Many approaches for decision trees construction have been proposed in the literature. However, most of them, focus on the algorithm and follows a completely automatic process for constructing a decision tree. In these approaches, only a few parameters are configured by the analysis expert before the start of the tree construction process. As they do not involve the expert in construction process itself, precious human knowledge is wasted, instead of being inserted on final model [3],[4]. This knowledge includes tacit domain and context knowledge that resides in the expert head and was gained over the years of experience in data gathering and analysis. This knowledge is not coded formally in the data so it cannot be incorporated automatically by machine learning algorithms.

This work describes an interactive structure for the construction of decision trees, which combine mechanisms for user intervention with traditional algorithms for decision tree construction [8]. This structure allows the intervention of the expert at any time during the construction of the tree. This enables the expert to verify and, if desired, direct on the choice of classification attributes and splitting points of the tree. This interactivity allows the expert to bring his domain and context knowledge to the tree construction process. In this user-centered approach, the expert and the computer can both contribute their strengths to the tree building process: the user providing domain knowledge and evaluating intermediate results of the algorithm, the computer automatically extracting patterns, satisfying the user constraints, and generating appropriate visualizations of these patterns.

For this approach to work, one needs effective interaction mechanisms between the machine and the experts. For that, we use visual data mining techniques [16], that combine powerful visualization techniques with interactive query mechanisms. In particular, we use a hierarchical visualization technique called Treemap [6],[18],[19], combined with graphical widgets for fast tree exploration and querying. These mechanisms allows the experts to visualize and explore the intermediate trees built by the J38 classification algorithm [12]. This structure allows human interaction to be performed with the computer during the whole classification process.

2. DECISION TREE CLASSIFIERS

Decision tree classifiers [8] learn a discrete-valued classification function, which is represented by a decision tree. Figure 1 shows an example of a tree for deciding if one can play outside or not. The tree has leaf and intermediate nodes. Each intermediate node corresponds to an attribute test. Edges symbolize all possible outcomes of the test in the node. The leaf node contains the label of one of the existing classes (in the example, yes or no). A path from the root to a leaf node defines a classification rule in the decision tree. In Figure 1, the left most path contains the rule: IF outlook=sunny AND humidity<=75 THEN yes (one can play outside).

The construction of the decision tree consists of two phases: (1) the construction itself; and, (1) the pruning of the tree. In the first phase, the tree is constructed by recursively partitioning the training set until each partition consists mostly of records of the same class. This set is then labeled as a leaf node. For each intermediate node, an attribute is selected. This attribute must not have been used yet in the classification path. The chosen attribute is the one that promotes the maximum segregation among the records with respect to the classification criteria. If the selected attribute is categorical, a sub-tree is created for each of its possible values. If the attribute is numerical, the algorithm verifies the value (or values) that better splits the data with respect to class segregation. A split (normally binary) is then created, with tests of the type "less than (<)" and "equal or greater than (≥)" the chosen split value.

The construction of the tree is guided by the objective decreasing the difficulty of classification. Thus, the choice of the attribute that will constitute the decision node, as well as the selection of the split point for numerical attributes, is carried through the maximization of the discrimination between the classified class [7],[8].

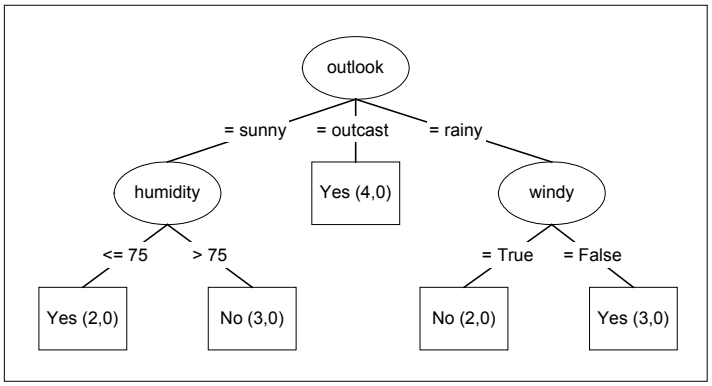


Fig. 1. A Decision Tree

Many times the resulting trees have very specialized rules. These trees adjust excessively to the peculiarities of the training set, and tend to produce deductive models of the training set instead of inductive models of the real world. On those trees, leafs nodes are supported by a small number of examples that represents isolated facts and that do not reflect reality. The pruning phase consists of simplifying the constructed tree to remove its excessively specialized parts. This generates less complex and more significant structures in terms of induction of the real world.

3. Interactively Building Decision Tree Classifiers

Many approaches for decision tree construction have been proposed in the literature. Most of them, however, focus on completely automated algorithms for tree construction, allowing only a few parameters to be configured before the start of the algorithm. Typically, the user involvement in this process is limited to the choice of the data and the parameterization of algorithm that will be used. Once the process of construction starts, the algorithm does not allow user intervention or the visualization of intermediate results, only of the final model is shown to the user [3].
Ankerst proposed an interactive model for decision tree construction [4]. In this model the user cooperates with the computer in the construction of the tree. The user can choose the next node to be expanded, select the attribute that will partition the data as well as its split points, or to leave those decision to the system

3.1 A Model for Interactive Decision Tree Construction

Figure 2 shows the schema that we adopt for an interactive session of a decision tree construction. When the session is started, the tool associates a notepad with it. The user can use this notepad to record comments, decisions, insights, or any other information he considers worth of registering. The notepad can be called at any time during the classification session. The start of the session is also when the user prepare data for the classification.

When the session ends, the user can, in accordance with his necessity, store or discard the tree produced up to that moment together with all notes made during the session.

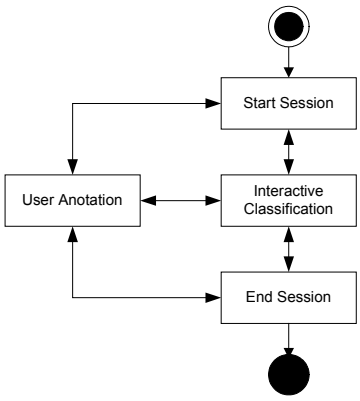


Fig. 2. A Classification Session

The central phase of a classification session is the interactive construction of the classification tree. For this phase, we adopted an adapted version of the interaction model proposed by Ankerst [4]. The adapted model is shown in Figure 3. This figure shows the activities performed by the user and by the computer. The activities performed by the user are shown in white rectangles and the activities performed by the computer are shown in gray rectangles.

The model is centered in the user. After the beginning of a work session, the user starts the interactive construction of the tree by visualizing and exploring the tree constructed up to that the moment. After exploring the tree, the user can choose one of the following operations:

- Manually remove a node of the tree. This corresponds to a manual pruning, where the current node is transformed into a leaf node and all its children are removed from the tree;
- Ask the system to suggest a list of possible classification attributes and splits points for a specific node. The attributes are ordered by its mathematical information gain for class segregation;
- Choose an attribute and its split point, and ask for the system to execute the expansion of a node based on this attribute;
- Ask for the system to explore data. In this case the user may be in doubt about which attribute choose, or the options must be researched for a more accurate choose;
- Ask for the system to automatically expand a node of the tree. In this case the attribute with bigger mathematical gain is automatically chosen by the system;
- Ask for the system to automatically expand a level of the tree. In this case all nodes in the current level are automatically expanded by the system;
- Ask to the system to prune the tree constructed up to that moment. In this case, the user has to input the parameters for pruning, and the system will automatically remove from the tree all nodes that it judges too specialized for effective classification.

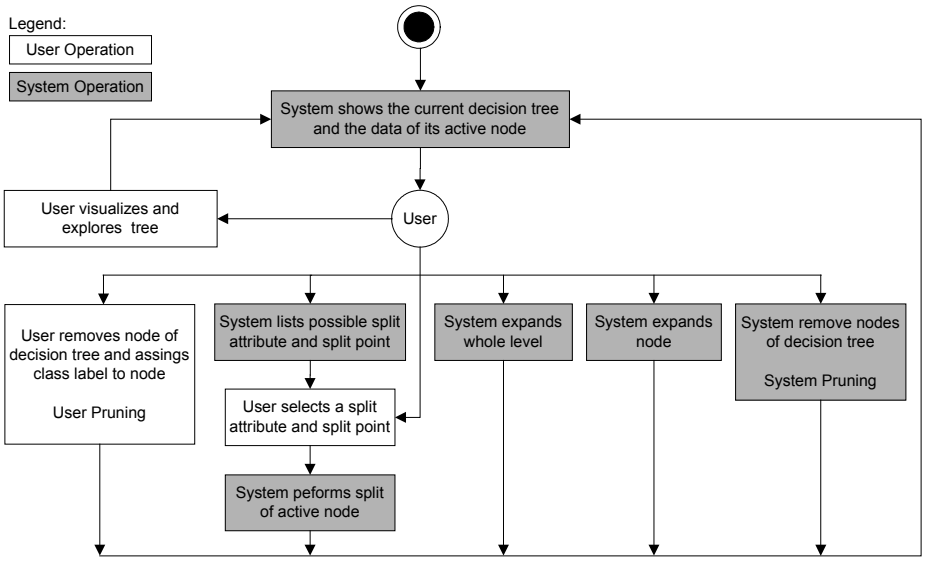


Fig. 3. Model for Interactive Decision Tree Construction

The model described in Figure 3 has some important differences from the model originally proposed by Ankerst. Our model allows the expansion of a complete level of the tree. It also allows to prune the tree at any time during its construction. Most important of all, the visualization and interactive exploration mechanisms are quite different from those proposed by Ankerst [5],[13], that is a pixel-based approach where each pair attribute-value is represented by one colored pixel in the visual screen. We propose the use treemaps [19] associated with query devices and details on demand controls that are typical from visual data mining tools [14],[16]. These resources are instrumental to the success of the model shown in Figure 3, they are intuitive and allow the user to explore the characteristics of these trees and the data associated with their nodes.

4. MINER - CLASSIFICATION TOOL

4.1 Preparation

The first step on the tool is the preparation for classification, we assume here that data preparation is done before, which may involve cleaning and data transformations. The tool supports the selection of attributes in case of data sets with large numbers of variables - performing some preliminary selection operations to bring the number of variables to a manageable range. These resources were inherited from Weka tool and help the user to identify the most relevant variables and determine the complexity and/or the general nature of models that can be taken into account in the next stage.

Data can be imported from a file in various formats: ARFF, CSV and can also be read from a URL or from an SQL database (using JDBC). The figure 4 shows the selection of some attributes to be used in the next stage.

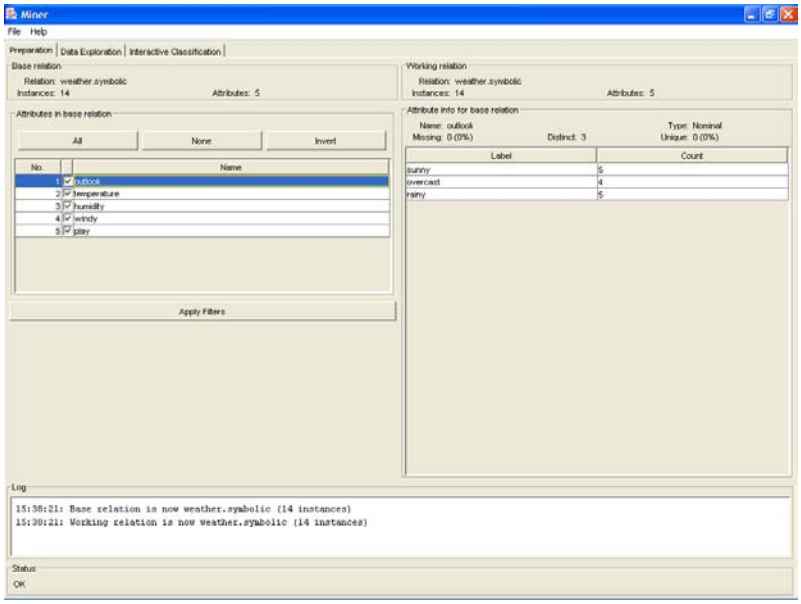


Fig 4. Preparation

4.2 Data Exploration

Treemaps is an information visualization technique proposed by Shneiderman for visualizing hierarchical structures [18],[19]. They use 100% of the available space for information visualization, mapping hierarchies in rectangular regions. The traditional representation of trees use lines to establish the connection between parents and children nodes of a hierarchy. This type of representation has two significant disadvantages: (1) a great portion of the available visual space is spent in the organization of the nodes; and, (2) non-trivial hierarchical structures generate trees of difficult visualization.

Because all the space available for drawing is used, Treemaps allow the efficient visualization of large hierarchies that can be in the order of thousands of items [18]. It is also very efficient in coding node attributes using the rectangles size and color. The visualization and exploration tools were inherited from TreeMiner, a tool that was developed at Salvador University [1],[2], it integrates treemaps with interactive queries and details on demand devices, offering an efficient mechanism for exploration of hierarchical data. The figure 5 shows the visualisation of an hierarchy of attributes .

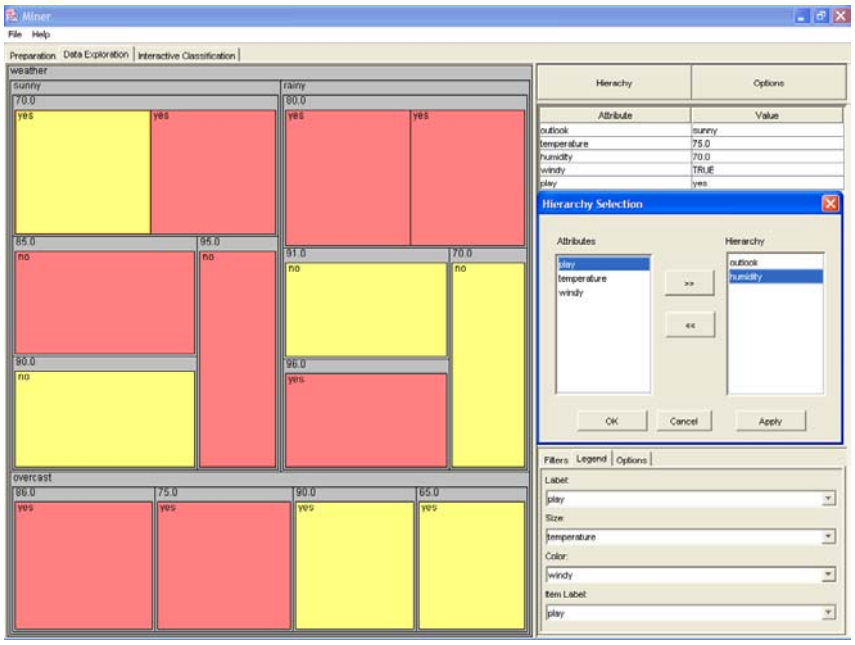


Fig 5. Data Exploration

4.3 Interactive Classification

The main part of the tool we developed is the interactive classification. The tool implements the model showed in the figure 6 (a,b,c and d) for interactive decision tree construction adapting the TreeMiner visualization structures, dynamic query and details on demand devices.

As illustrated in the Figure 3 model, the visualization of intermediate trees is brought up to date at each step of the tree construction process, allowing the expert to explore it, and re-direct the tree building process if necessary.

In the case of decision trees, the visual attribute "size" is especially indicated for representing the number of records that supports a node. The visual attribute "color" is especially indicated to represent node's purity degree – by which all records of a node belongs to the same class. Each class (e.g., yes and no) can be associated with a distinct color (e.g., red and yellow). The purity degree of a leaf node can be visualized by the cleanness of the rectangle color representing it. This creates an intuitive representation for the decision tree nodes support and purity at any point of its construction..

Figure 6 (d) shows the decision tree of Figure 1 represented as a treemap. This representation not only shows the values of the classifying attributes, but it also shows the purity and support of its leaf nodes. As hinted before, the purity of the node is represented through the colors, and the number of records that supports the nodes is represented through the size of the rectangles. Thus, the treemap offers a view of the decision tree and the class distribution of the mined base.

Beyond the visualization, the developed tool supplies interaction mechanisms based in visual data mining techniques to assist the user to explore the produced tree. Upon selecting a node in the tree, the information about that node is shown in right superior corner of the screen, see Figure 6. This shows the following to the user: the dominant class of the node, the amount of registers that supports the node, and the total number of impurities in the node (i.e., the number of records that do not belong to the node dominant class). Moreover, if necessary, the user can also request the list of all the registers that support the selected node.

The tool also has, see bottom right corner of Figure 6, filters that are typical of visual data mining. The "depth filter" hides deeper nodes of the tree, allowing to the user to quickly examine simplified trees, and take pruning decisions. The class filter redraw the tree using only the selected classes. The "impurity filter" allow the interactive verification of which nodes have impurity values over and under the set values. And, the "number of records filter" allow the verification of which nodes have support over and under the set values. It is important to observe that the time between user interaction with these filters and the tool updating of the visual screen is practically zero. With these interaction resources the user can quickly decide which node to remove or to expand, or if the obtained result is already satisfactory.

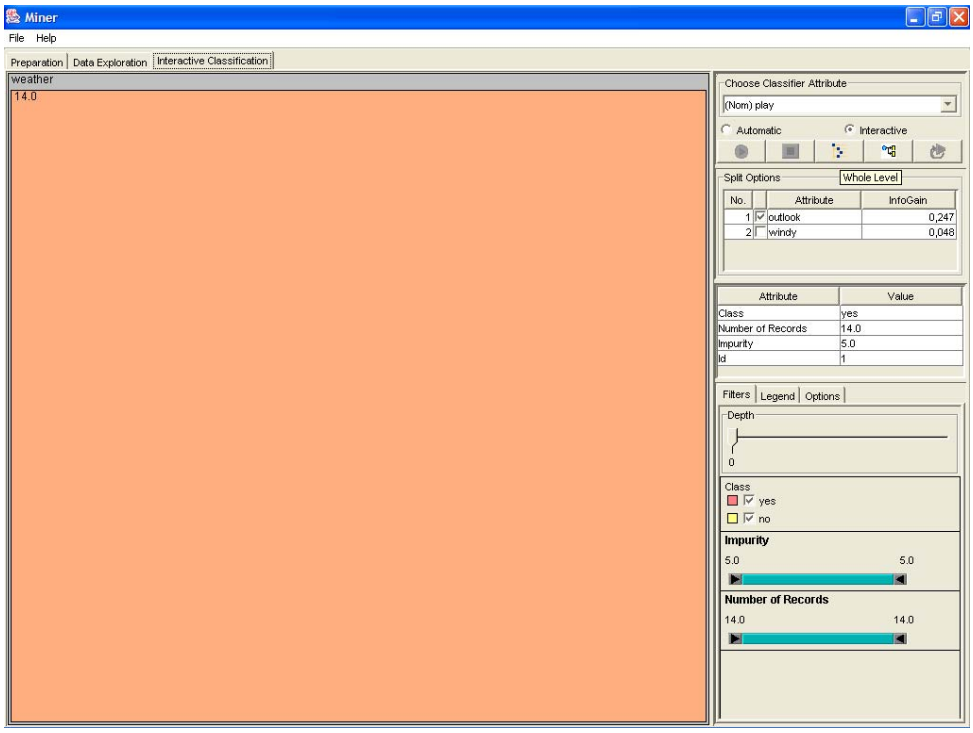


Fig 6 (a). Beginning of the Interactive Classification

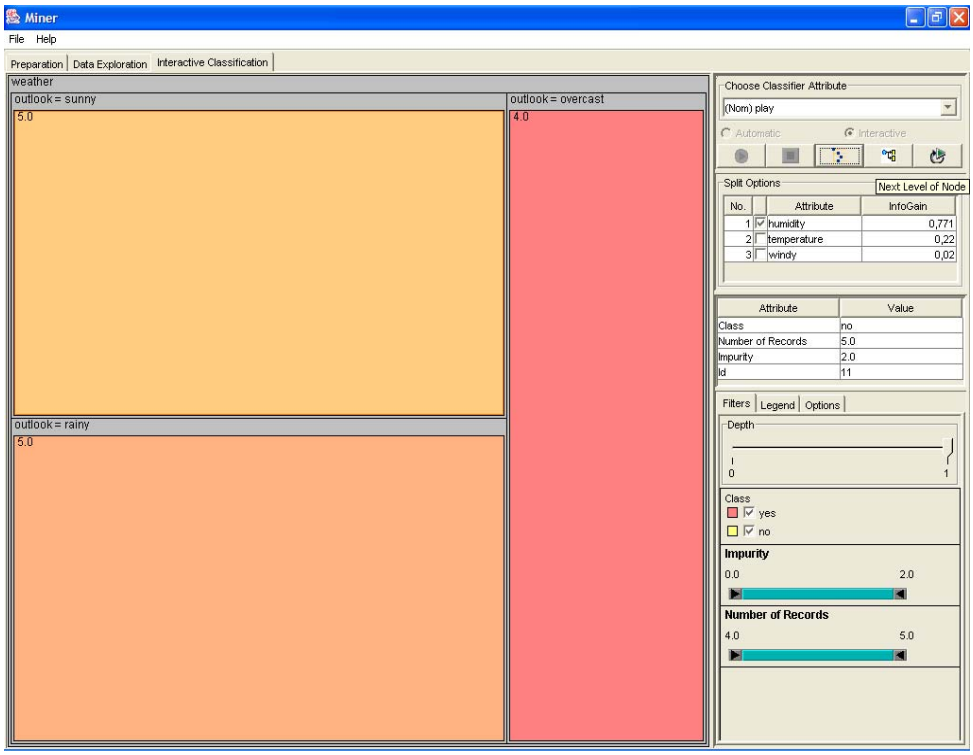


Fig 6 (b). Interactive Classification – Expand a whole level

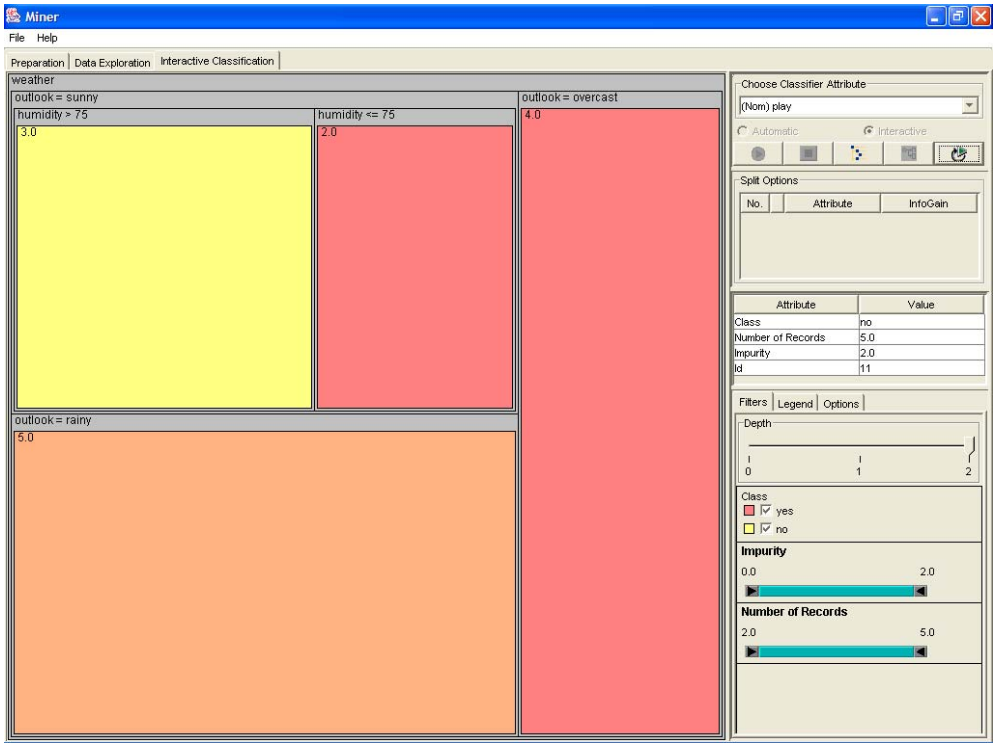


Fig 6 (c). Interactive Classification – Expand a selected node of the tree

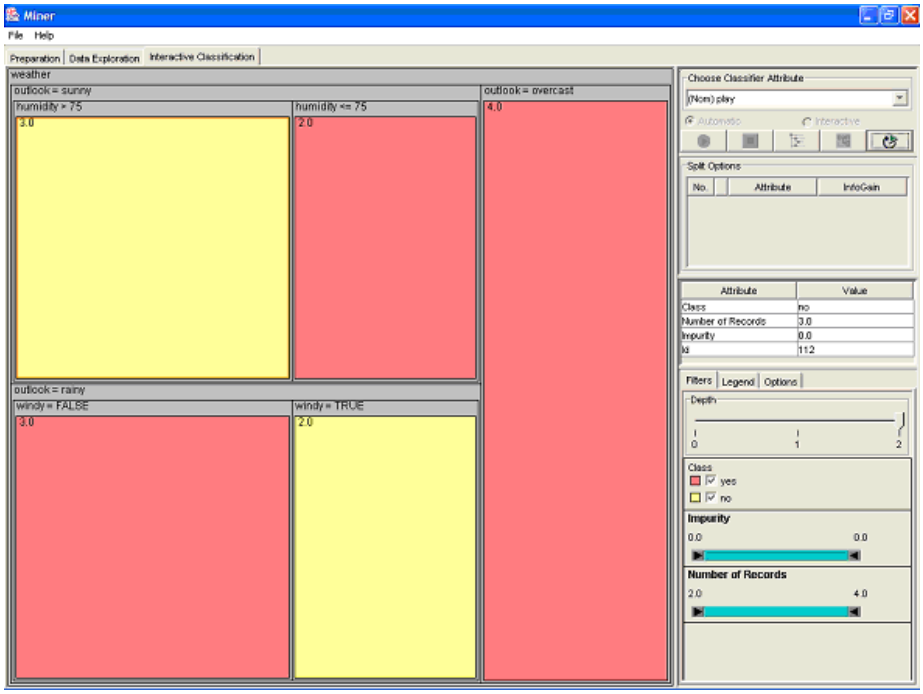


Fig 6 (d). The decision tree

5. Some Implementation Details

The tool that implements the interaction model described in Figures 2 and 3 was developed from two existing tools: (1) Weka, an environment for knowledge discovery, that makes possible, among others things, the construction of classification trees; and (2) TreeMiner, a treemap based hierarchical visual data mining tool. The two tools are implemented in Java and are public domain, this facilitated their integration, adaptation, and expansion.

The Weka Tool (Waikato Environment will be Knowledge Analysis) was developed at the University of Waikato in New Zealand [10],[12]. It implements algorithms for several data mining techniques, including the J48, an improved version C4.5 for building decision trees [17]. The Weka is available in following web address: http://www.cs.waikato.ac.nz/ml/weka/.

TreeMiner was developed at Salvador University [1],[2], it integrates treemaps with interactive queries and details on demand devices, offering an efficient mechanism for exploration of hierarchical data. The tool we developed for interactive decision tree construction adapts the TreeMiner visualization structures, dynamic query and details on demand devices, as described in Section 4 and portrayed in Figure 4.

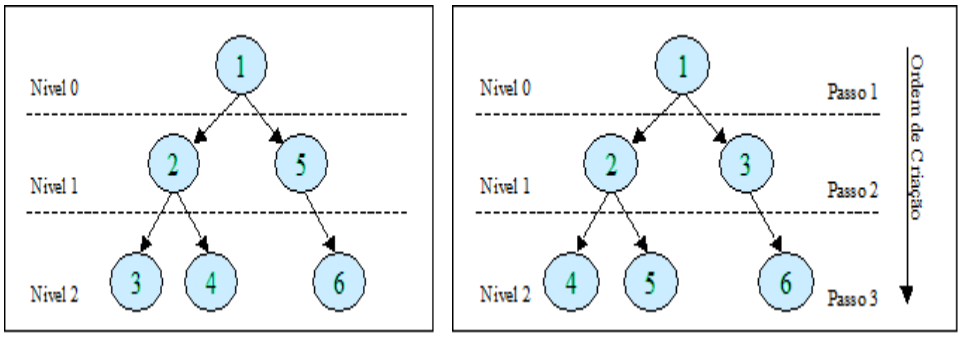


Fig. 7. Old and New Sequence of Tree Nodes Construction

From Weka tool, the tool we developed uses the structure for creating classification sessions, its resources for data preparation, its resource for algorithm parameterization and the base of the J48 algorithm for constructing classification trees. The algorithm, however, had to be adapted in order to make it possible the interaction with the user during the construction of the tree. It was adapted to allow: (1) the construction of the tree level by level, (2) the

construction of the tree node by node, and (3) listing the attributes and its splitting points for a given node. For the first functionality, the algorithm was modified to function level by level (breadth first) instead of recursively (depth first). This way, the user can interact better with the algorithm during the tree construction. Figure 7 illustrates the old and the new sequence of node construction carried by the algorithm.

Besides the cited adaptations in the Weka and the TreeMiner tools, the following functionalities were or are being added to our decision tree construction tool: (1) a functionality for user annotation is available during the whole tree construction process; (2) filters are being built to assist the selection of data for the training sets; (3) a functionality for mapping numerical attributes into categorical ones is being integrated into the tool; and , (4) the parameterization of the J48 algorithm is being modified to accept the assignment of weights for the attributes.

6. CONCLUSION

The current classification approaches allow a limited participation of the expert during the decision tree construction process. These approaches do not take advantage of the experts´ knowledge with respect to the data and domain being mined. With a cooperative classification approach, the expert and the computer can contribute with their best. The computer providing the capacity to recognize mathematical patterns. The experts providing their ability to interpret and promote a deeper understanding of these patterns. This combination should generate more trustworthy models.

This work defined a model and implemented a tool for the interactive construction of classification trees. The defined model is adapted from the cooperative classification model originally proposed by Ankerst. The implemented tool enacts this model, using treemap visualizations and visual data mining query devices to create efficient mechanisms for supporting user-computer interactions during the construction of decision trees.

The tool was developed by adapting a visual data mining tool, TreeMiner, and an environment for knowledge discovery, Weka. The resulting tool supports all phases of classification process, from the data selection to the interactive exploration of the obtained results.

As future work, we intend to include new mechanisms for visualizing and interacting with the decision trees. In particular, we intend to add a module that uses traditional tree drawing methods to visualize and interact with produced trees. This will allow the comparison of the treemap approach with more traditional approaches for tree visualizations and exploration. We also intend to incorporate a module for visually comparing generated trees by overlapping them on the computer screen at each iteration of the tree construction process.

References

[1] Almeida, M.: A Tool for Visual Data Mining Using Treemaps and its Applications (in Portuguese). Master of Computer Science Thesis, Salvador University (UNIFACS), Salvador, Ba, Brazil, (2003).

[2] Almeida, M.; Mendonça Neto, M. Using Treemaps to Internalize Knowledge (in Portuguese). In the Proceedings of the First Brazilian Workshop on Information Systems and Knowledge Management, SBC, Fortaleza, CE, Brazil, 2003. v. 1, p. 1-11.

[3] Ankerst, M., Elsen, C., Ester, M., Kriegel, H.-P.: Visual Classification: An Interactive Approach to Decision Tree Construction. In ACM SIGKDD 5th Int. Conf. On Knowledge Discovery and Data Mining, San Diego, CA, USA, pp. 392-396 (1999).

[4] Ankerst, M., Ester, M., Kriegel, H.-P.: Towards an Effective Cooperation of the User and the Computer for Classification. In ACM SIGKDD 6th Int. Conf. On Knowledge Discovery and Data Mining (KDD 2000), Boston, MA, pp. 179-188 (2000).

[5] Ankerst, M.: Visual Data Mining with Pixel-Oriented Visualization Techniques. Proc. Workshop Visual Data Mining (2001).

[6] Babaria, K.: Introduction to Treemap. University of Maryland. [on line] http://www.cs.umd.edu/hcil/treemap3/TreemapIntroduction.pdf (2001).

[7] Barlow, T., Neville, P.: Case Study: Visualization for Decision Tree Analysis in Data Mining. Proc. of IEEE Symposium on Information Visualization - INFOVIS´01 (2001).

[8] Breiman, L., Friedman, J., Olshen, R., and Stone, C.: Classification and Regression Trees. Belmont, CA: Wadsworth (1984).

[9] Fayyad, U., Piatetsky-Shapiro, G., Smyth, P.: From Data Mining to Knowledge Discovery in Databases. In AI Maganize, pp. 37-54 (1996).

[10] Frank, E. and e. al: Weka [http://www.cs.waikato.ac.nz/ml/weka/], The University of Waikato.

[11] Garner, S.R.: ARFF-the WEKA dataset format. World Wide Web hypertext document at http://www.cs.waikato.ac.nz/~ml/weka/arff.html.

[12] Garner, S.R.: WEKA: The Waikato Environment for Knowledge Analysis. In Proc. of the New Zealand Computer Science Research Students Conference, pages 57—64 (1995).

[13] Keim, D. A.: Designing Pixel-Oriented Visualization Techniques: Theory and Application. In IEEE Trans. on Visualization and Computer Graphics, vol 6: IEEE Computer Society, pp. 59-78 (2000).

[14] Keim, D. A.: Information Visualization and Visual Data Mining. In IEEE Trans. on Visualization and Computer Graphics, vol 7, nº 1, pp. 100-107 (2002).

[15] Mendonça Neto, M.; Sunderhaft, N. A State of the Art Report: Mining Software Engineering Data. State of the Art Technical Report DACS-SOAR-99-3. U.S. Department of Defense (DoD) Data & Analysis Center for Software, Rome, NY, 1999. Also available in: http://www.dacs.dtic.mil/techs/datamining/datamining.pdf

[16] Oliveira, M., Levkowits, H.: From Visualization to Visual Data Mining: A Survey. In IEEE Transactions on Visualization and Computer Graphics, vol 9, nº 3, pp. 378-394 (2003).

[17] Quilan, J. R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Mateo, CA (1993).

[18] Shneiderman, B., Johnson, B.: Treemaps: A Space-Filling Approach to the Visualization of Hierachical Information Structures. Proc. of IEEE Information Visualization, pp. 275-282 (1991).

[19] Shneiderman, B. Tree visualization with tree-maps: 2-d space-filling approach. ACM Transactions on Graphics, vol. 11 , nº. 1, pp. 92-99 (1992).

Segmentación de Imágenes de Rango por Detección de Bordes Empleando un Algoritmo Genético

Idanis Diaz
Universidad de Medellín, Facultad de Ingeniería de Sistemas
Medellín, Colombia
idiaz@guayacan.udem.edu.co

and

John W Branch
Universidad Nacional de Colombia, Escuela de Sistemas
Medellín, Colombia
jwbranch@unalmed.edu.co

and

Flavio Prieto
Universidad Nacional de Colombia, Departamento de de Ingeniería Eléctrica, Electrónica y Computación
Manizales, Colombia
fprieto@nevado.manizales.unal.edu.co

Abstract

The following article presents a images segmentation method for partitioning surface in range image into planar patches. This method is based on edges map detection by calculations of depth gradients and orientation gradients and a genetic algorithm. The objective is to delimit the planar patches contained in images to facilitate the labeling of each region. The genetic algorithm is guided by depth gradients and orientation gradients in order to find the edges map.
Keywords: Range Image, Segmentation, Genetic Algorithm, Edges.

Resumen

El siguiente artículo presenta el desarrollo de un método de segmentación para imágenes de rango de superficies planas, basado en la detección de un mapa de bordes por medio del cálculo de gradientes de orientación, gradientes de profundidad y un algoritmo genético. El método de segmentación consiste en delimitar las superficies de la imagen, facilitando el etiquetamiento de los píxeles que pertenecen a cada región. La tarea desempeñada por el algoritmo genético consiste en conformar bordes delgados, no fragmentados a partir de los gradientes calculados.
Keywords: Imagen de Rango, Segmentación, Algoritmos Genéticos, Bordes

1. INTRODUCCIÓN

El problema de la reconstrucción computacional de formas tridimensionales, ha recibido una enorme atención en investigaciones de visión artificial en la última década. El interés surge debido a que la teoría de formas tiene aplicaciones en una amplia variedad de campos, como por ejemplo: el diseño, la automatización de manufacturas, el mapeo de terrenos, la robótica, entre otros.

Las formas de las superficies para la reconstrucción en tres dimensiones, son bien representadas con un tipo especial de imágenes, llamadas imágenes de rango, cuya principal característica, es que cada píxel es una medida de la distancia entre un punto visible de la escena y un marco de referencia conocido; en lugar del nivel de intensidad de un color, como sucede con las imágenes de intensidad.

Existe una variedad de métodos para la reconstrucción de formas o superficies a partir de imágenes de rango, la mayoría de ellos están conformados por un conjunto de etapas o tareas que consisten en la adquisición de datos, etapa encargada de la captura de un conjunto de medidas de profundidad, a partir de varias vistas del objeto. El registro e integración, etapa encargada de unir múltiples vistas y llevarlas a un sistema de coordenadas común. La segmentación, que consiste en agrupar puntos con propiedades homogéneas en regiones etiquetadas y por último, el ajuste de superficies, que consiste en estimar una superficie paramétrica a las diferentes regiones obtenidas con la segmentación.

Cada una de las etapas de la reconstrucción de superficies independientemente se ha constituido en el objetivo de diversos trabajos de investigación, que han tenido como finalidad el establecer técnicas o métodos que logren un desempeño óptimo y generalizado para cada una de ellas. El presente artículo se concentra en una de estas etapas, la etapa de la segmentación para imágenes de rango de superficies planas.

La mayoría de técnicas de segmentación propuestas para imágenes de rango se basan en el principio de crecimiento de regiones, otras se basan en el principio de la detección de bordes o de formas, en la imagen. Los métodos basados en el principio de crecimiento de regiones tienen el problema de dejar mal definidos los bordes entre las regiones. Por su parte, los métodos basados en el segundo principio, presentan la dificultad de que las técnicas empleadas para detectar bordes por lo general entregan como resultados mapas de bordes fragmentados, gruesos y con falsas detecciones; como es el caso del método basado en la detección de cambios o gradientes de profundidad y orientación entre las superficies de la imagen.

En este artículo se propone un método de segmentación para imágenes de rango de superficies planas, que se especializa en encontrar un mapa de bordes empleando una técnica de búsqueda robusta como lo es un algoritmo genético. La función del algoritmo genético es añadir o desechar píxeles de una aproximación inicial de los bordes, obtenida a partir del cálculo de gradientes de orientación y profundidad, en búsqueda de un mapa de bordes sin fragmentaciones, delgado y sin falsos bordes, que delimite las superficies de la imagen de rango.

Un algoritmo genético, es una técnica de búsqueda estocástica basada en el principio de la evolución natural, en la que un conjunto de posibles soluciones de un problema se representa como una población de individuos que se transforman por medio de unos mecanismos de reproducción y mutación, generándose nuevos individuos o nuevas posibles soluciones. Las nuevas posibles soluciones se evalúan con una función de aptitud que guía la búsqueda hacia los individuos más óptimos. Cada vez que son creados nuevos individuos, se incorporan en la población y se considera que se ha pasado a una nueva generación, que también se somete a reproducción y mutación, hasta que se cumpla una condición de parada [7].

El problema de la detección de bordes en imágenes de rango se puede ver como un problema de búsqueda, que consiste en encontrar un conjunto de píxeles bordes que mejor delimite las superficies de la imagen. Una vez, delimitada las superficies se puede efectuar un rápido etiquetamiento de las regiones definidas por los bordes.

El artículo se encuentra organizado de la siguiente manera: en la sección 2 se hace una introducción al problema de la segmentación. En la sección 3 se describe el método de segmentación que se propone. En la sección 4 se presentan los resultados obtenidos al segmentar treinta imágenes de rango de superficies planas disponibles en la página de la Universidad del Sur de la Florida (http://marathon.csee.usf.edu). En la sección 5 se discuten los resultados obtenidos, y por último en la sección 6 se concluye el artículo.

2. EL PROBLEMA DE LA SEGMENTACION

El problema de la segmentación es un problema clásico que se puede definir de la siguiente manera [3]:

Si R es una imagen de rango completa, la segmentación consiste en dividir R en subregiones: $R_1, R_2, ..., R_n$, tal que:

- $U_{i=1}^{n} R_i = R$
- R_i es una región conectada para $i = 1,2,...,n$ y
- $R_i \cap R_j = \phi$ para todo i y j, si $i \neq j$
- $P(R_i) = TRUE$ para $i = 1,2,...,n$ y
- $P(R_i \cup R_j) = FALSE$ para $i = 1,2,...,n$

En donde $P(R_i)$ es un predicado lógico o un conjunto de propiedades similares sobre los puntos de R_i y ϕ es el conjunto nulo.

La segmentación de una imagen de rango consiste en colocar una etiqueta igual a los píxeles con propiedades geométricas similares o que pertenezcan a una misma superficie. Generalmente, las propiedades geométricas que se tienen en cuenta para la segmentación son: la normal, la profundidad, y las curvaturas media y gaussiana de cada punto en la imagen. Se han realizado muchas propuestas para segmentar imágenes de rango, algunas de ellas realizan el etiquetamiento a partir del crecimiento de una pequeña región o regiones de píxeles, llamadas regiones semillas, a la cual se le van añadiendo otros píxeles a su alrededor con las mismas propiedades geométricas, hasta que no se encuentren más píxeles que añadir. Otros métodos, en cambio, hacen una detección inicial de bordes o de formas en la imagen, para luego etiquetar los píxeles guiándose por los bordes o formas detectadas.

Entre los métodos de segmentación que trabajan bajo el principio de crecimiento de regiones se puede mencionar: el método de segmentación propuesto por la Universidad del Sur de la Florida (USF) que consiste en realizar un crecimiento de regiones a partir de un píxel semilla [3]. El método propuesto por la Universidad Estatal de Washington (WSU) que realiza un agrupamiento utilizando un algoritmo para cluster [3]. El método de segmentación propuesto por la Universidad de Edimburgo (UE) basado en el cálculo de curvaturas medias y gaussianas [3]. El método de segmentación propuesto por la Universidad de Bern (UB) que hace un crecimiento de regiones agrupando segmentos de líneas en vez de puntos [4]; y el método de Besl y Jain que sigue el principio de crecimiento de regiones para detección de superficies curvas y planas y ajusta superficies de orden variable [1].

Entre los métodos de segmentación que trabajan bajo el principio de la detección de bordes o de formas en la imagen, el método de segmentación propuesto por la Universidad de Osaka (OU) que consiste en hacer un agrupamiento de píxeles guiado por información extraída de los bordes de la imagen [6]. El método propuesto por la Universidad de Algarbe (UA) agrupa píxeles según su profundidad y segmenta por niveles guiándose por una detección de bordes [6] ; y el método propuesto por la Universidad Pontificia de Paraná (UFPR) el cual realiza un crecimiento de regiones guiado por los bordes de la imagen [10].

En trabajos como: [3], [6] y [8], se presentan y comparan los resultados obtenidos al evaluar algunos de los métodos de segmentación de imágenes de rango, mencionados anteriormente, utilizando la metodología de evaluación propuesta por Hoover [3]. Esta metodología de evaluación consiste en comparar píxel por píxel con una tolerancia, la segmentación obtenida por el método contra una imagen segmentada manualmente. Los resultados obtenidos demuestran que los segmentadores hasta el momento tienden a incorporar ruido, a detectar menos instancias correctamente y a la perdida de regiones, a medida que se les exige más precisión en la evaluación, aumentando la tolerancia.

Los métodos de segmentación basados en el crecimiento de regiones, tienen como desventaja que su eficiencia depende mucho de las regiones iniciales que se tomen como semilla, además, el crecimiento de regiones sin una previa guía de los limites, conduce a bordes mal definidos. Por otra parte, los métodos basados en la detección de bordes o de formas, se enfrentan con el problema de que encontrar bordes o formas en una imagen de rango no es una tarea fácil. Por ejemplo, en el caso de la detección de bordes, muchas técnicas, entregan como resultados bordes gruesos, incompletos, falsos, entre otros.

3. EL METODO DE SEGMENTACION PROPUESTO

El método de segmentación que se propone, está constituido por tres etapas, en la primera de ellas se calculan gradientes o cambios de profundidad y de orientación; en la segunda etapa se realiza la búsqueda de un mapa de bordes que delimite las superficies de la imagen, a partir de los gradientes calculados, utilizando un algoritmo genético y por último se lleva a cabo un rápido etiquetamiento de los píxeles encerrados por los bordes de la imagen. A continuación se explicaran cada una de estas etapas.

3.1. Calculo de Gradientes

Para el cálculo de los gradientes de profundidad y orientación se utilizó el procedimiento presentado en [9], que consiste en calcular cambios entre los valores de profundidad y cambios entre las normales en cada punto de la imagen.

Los procedimientos seguidos para el cálculo de los gradientes son:

Cálculo de Gradientes de Profundidad

Los gradientes de profundidad, corresponden a las intersecciones entre regiones, cuyos píxeles presentan una variación significativa en su coordenada *Z*. Para obtener estas variaciones de profundidad significativas, se calcula para cada punto de la imagen la diferencia de profundidad con respecto a sus puntos vecinos, en una ventana de N X N.

La diferencia de profundidad para cada píxel es:

$$D_p(i, j) = \max(|w_{i,j} - W|) \quad (1)$$

Donde, $w_{i,j}$ es el valor de Z correspondiente a las coordenadas 3D del píxel *i,j* y W es el valor de Z de los puntos vecinos.

Luego de obtener las diferencias de profundidad, se seleccionan, como gradientes de profundidad aquellos píxeles o puntos de la imagen que estén por encima de un umbral.

Cálculo de los Gradientes de Orientación

Los gradientes de orientación, corresponden a las intersecciones entre superficies, que presentan un cambio de dirección entre sus vectores normales. Este procedimiento se puede dividir en dos etapas:

a) Cálculo de vectores normales para cada punto.

A cada punto de la imagen se le calcula un vector normal, con la ayuda de una ventana de K X K, de esta ventana se toman cuatro puntos en las direcciones (Norte, Este, Sur y Oeste), con los cuales se obtienen cuatro vectores: V_N, V_E, V_S, V_O , partiendo desde el píxel central.

Dado dos puntos en el espacio, un vector entre ellos se puede calcular de la siguiente forma:

El vector que va del punto $P(p_1, p_2, p_3)$, al punto $Q(q_1, q_2, q_3)$, es: $V = (q_1 - p_1, q_2 - p_2, q_3 - p_3)$ (2)

El vector normal para cada punto se obtiene utilizando la siguiente ecuación:

$$\vec{N}_{i,j} = \frac{\sum(\vec{V}_D \otimes \vec{V}_{nextD})}{4} \quad (3)$$

D = { Norte, Este, Sur, Oeste}

nextD, es la próxima dirección a D en sentido horario.

Es decir, el vector normal del punto (i,j), es una media del producto vectorial de cuatro vectores, tomados de dos en dos dentro de una ventana de K x K.

Al tomar los puntos para el cálculo de los cuatro vectores, se tiene en cuenta que, entre el píxel central y el respectivo punto en el extremo de la ventana, no exista una variación de profundidad significativa, es decir, un gradiente de profundidad.

b) Cálculo de dirección entre los vectores normales

Con una ventana de L X L, se obtiene la máxima diferencia angular de cada punto de la imagen, con respecto a sus vecinos. Los ángulos entre vectores se calculan de la siguiente forma:

$$\theta_{kl} = arc\left(\frac{\left\|\vec{N}_{i,j}\right\|\cdot\left\|\vec{N}_{k,l}\right\|}{\vec{N}_{i,j}\bullet\vec{N}_{k,l}}\right) \quad (4)$$

$N_{k,l}$, son los vectores normales de los puntos vecinos a *i,j*.

$\theta_{k,l}$, es el ángulo formado entre los vectores $N_{i,j}$ y $N_{k,l}$.

Se toma como diferencia angular del punto central de la ventana de L x L, al máximo valor de θ obtenido: $D_o(i,j) = \max(\theta_{k,l})$, si la diferencia angular del punto central, es mayor que un umbral establecido.

3.2. Búsqueda del Mapa de Bordes

Los gradientes de profundidad definen bordes gruesos; pero bastante cercanos a los verdaderos bordes de profundidad. Por su parte, los gradientes de orientación definen bordes gruesos, interrumpidos y mal ubicados. Como se puede observar en la Figura 1.

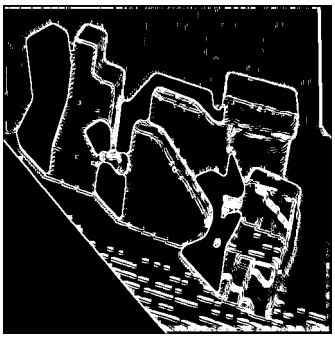


Figura 1. Bordes definidos por los gradientes

La búsqueda del mapa de bordes se lleva a cabo por medio de un algoritmo genético. El algoritmo genético se encarga de seleccionar píxeles gradientes que conformen bordes delgados, desechar píxeles gradientes que conformen falsos bordes y completar bordes interrumpidos añadiendo los píxeles necesarios.

Al implementar un algoritmo genético se deben definir, un esquema de representación para codificar cada posible solución del problema como un individuo de la población a evolucionar, un mecanismo para generar la población inicial, un mecanismo de reproducción y otro de mutación, una función de aptitud para medir que tan cercano está cada individuo del óptimo deseado, un criterio de parada que permita que el algoritmo termine de iterar cuando se cumpla una condición y unos parámetros de control como son: el número de individuos de la población, y el porcentaje de los individuos a mutar y a reproducir en cada generación. Cada uno de estos elementos se describe a continuación:

El esquema de representación. El esquema de representación, consiste en un arreglo bidimensional binario, del tamaño de la imagen de rango a segmentar. Cada individuo representa un mapa de bordes, en los que el valor uno (1), indica que el píxel correspondiente es un borde en la imagen y el valor cero (0), que no lo es.

Generación de la Población Inicial. La población inicial de individuos, se obtiene a partir de los bordes definidos por los gradientes de profundidad, los cuales son gruesos y fragmentados; pero bien ubicados. A los bordes definidos por los gradientes de profundidad se les aplica un procedimiento tradicional de adelgazamiento, se copian en cada individuo e inmediatamente se les aplica el operador de mutación para variar sus configuraciones.

Operador de mutación. Este operador es el encargado de seleccionar puntos gradientes al azar o puntos bordes terminales, para expandirlos, uniéndolo con otros puntos gradientes o bordes, en una dirección determinada.

La operación de mutación se realiza de la siguiente manera:

1) Se selecciona al azar un píxel candidato o un punto inicial para ser extendido, cumpliendo con una de las siguientes condiciones.

   a) Un punto terminal, perteneciente a un borde interrumpido, en el individuo.
   b) Un píxel borde de profundidad en el individuo, sin vecinos, es decir, aislado.
   c) Un píxel no perteneciente a ningún borde en el individuo, que corresponda con la ubicación de un gradiente de orientación.

2) Se escoge un punto objetivo cercano al candidato inicialmente seleccionado, para unirlos, trazando un segmento de borde entre ellos.

   Los puntos escogidos como objetivos, deben estar por lo menos a una distancia mínima de tres píxeles con respecto al punto inicial y a una distancia máxima determinada por una ventana de 13 x 13, colocada alrededor del punto inicial.

   Para seleccionar un punto objetivo, se tienen cuatro opciones:

   a) El píxel borde más cercano al punto inicial, con una distancia mínima de dos píxeles.
   b) Un punto borde o un píxel correspondiente con algún gradiente de orientación, que se encuentre linealmente con respecto al borde que se este extendiendo.
   c) Un punto que corresponda a un gradiente de orientación, seleccionado al azar, que se encuentre alrededor del punto inicial.
   d) El punto con la máxima diferencia entre normales del vecindario de 13 x 13, sin corresponder a un gradiente de orientación; es decir, sin ser mayor al umbral de orientación establecido.

   Si el punto inicial a ser extendido es un píxel borde de profundidad, preferencialmente se selecciona como objetivo el píxel borde más cercano del vecindario; si no hay un píxel borde cercano, se prefiere un punto que satisfaga la condición de la opción b, si la condición no se satisface, se selecciona un punto que satisfaga la condición c; si no se cuenta con puntos gradientes de orientación alrededor, entonces se escoge la última opción.

   De igual forma, si el punto inicial a ser extendido corresponde a un gradiente de orientación, preferiblemente se toma como punto objetivo la opción b, si no se presenta el caso de la opción b, se selecciona al azar un punto correspondiente a un gradiente de orientación o el punto borde más cercano; si no se cuenta con un píxel que cumpla las condiciones de las opciones a y c, finalmente se escoge la opción d.

3) Una vez se ha trazado un primer segmento de borde entre un punto inicial y un punto objetivo, se procede a tomar como punto inicial al punto objetivo, nuevamente se busca un punto objetivo que satisfaga las condiciones de las cuatro opciones mencionadas, y se traza un segmento de borde entre ellos. Este proceso se repite, hasta que el punto objetivo escogido, resulta ser un punto borde previamente seleccionado y marcado en el individuo.

Operador de cruce. Se utiliza el operador de cruce tradicional con un solo punto de corte.

Función de aptitud. La aptitud que se le asigna a cada individuo, favorece a aquellos que poseen píxeles bordes ubicados en la intersección de dos regiones distintas, que no contienen píxeles interrumpidos, ni tampoco pequeñas regiones definidas por los bordes encontrados. La función de aptitud es la siguiente:

$$Aptitud = WC\left(\frac{Pixeles\_Bordes\_Añadidos\_Correctamente}{Total\_Pixeles\_Bordes\_Añadidos}\right) + WPR\left(\frac{1}{No\_Pequeñas\_\mathrm{Re}giones}\right) + WF\left(\frac{1}{Pixeles\_Ter\min ales}\right) \quad (6)$$

Total_Píxeles_Bordes_Añadidos: es el número total de píxeles bordes añadidos por el algoritmo genético.

Píxeles_Bordes_Añadidos_Correctamente: es el número de píxeles añadidos, que se encuentran en medio de dos regiones con normales distintas. Esto se determina observando los vectores normales de los píxeles pertenecientes a las dos regiones definidas por cada píxel borde más sus píxeles bordes vecinos, en una ventana de M x M.

No_Pequeñas_Regiones: es el número de pequeñas regiones conformadas por los bordes encontrados con el algoritmo genético.

No_Píxeles_Terminales: es el número de píxeles ubicados en los extremos de los bordes interrumpidos.

WC, WPR, WF, son pesos asignados a cada uno de los tres criterios de evaluación mencionados, respectivamente.

Criterio de parada. El algoritmo genético detiene la búsqueda cuando alcanza el máximo número de generaciones.

Parámetros. Luego de diferentes ensayos con distintos tamaños de población, número de generaciones y tasas de cruzamiento y mutación, se decidió ejecutar el algoritmo con 30 individuos, 50 generaciones, una tasa de cruzamiento de un 20% y una tasa de mutación de un 40%. Al fijar estos parámetros se tuvo en cuenta que el tiempo de respuesta del algoritmo no fuera muy alto y que no se desmejorara la convergencia del procedimiento.

El porcentaje de mutación establecido es mayor que el de cruce, por ser el operador de mutación el encargado de generar nuevos bordes en los individuos y por que el operador de cruce puede producir bordes interrumpidos al intercambiar el contenido de los individuos para generar los descendientes.

3.3. Etiquetamiento de regiones

Una vez, el algoritmo genético entrega un mejor mapa de bordes que delimita las regiones de la imagen, se procede a asignar etiquetas (un valor numérico) a los píxeles enmarcados en cada región. Esto se realiza asignándole una misma etiqueta a los píxeles conectados, que no estén separados por los bordes localizados. De esta forma cada región queda enumerada con un valor entero mayor o igual que 10; excepto aquellas que resultan muy pequeñas, con un número de píxeles inferior a 500, las cuales se etiquetan con un valor igual a 0. Fue necesario etiquetar las regiones delimitadas por los bordes con valores mayores o iguales a 10, porque a si lo requiere el algoritmo de evaluación propuesto en [3], que se encuentra implementado en: http://marathon.csee.usf.edu.

Al igual que las regiones pequeñas, las regiones definidas como sombras de la imagen, cuyos píxeles poseen valores de rango iguales a cero, y los bordes de la imagen, también fueron etiquetadas con un valor igual a 0. Los bordes de la imagen, límites de las regiones segmentadas, no se añadieron a las respectivas regiones, por considerar que su existencia, no afecta el cálculo de las ecuaciones paramétricas de las superficies, que se debe llevar a cabo en la etapa siguiente del proceso de reconstrucción de superficies 3D (ajuste de superficies). En el Anexo 1. se pueden observar varios ejemplos de una imagen etiquetada, con diferentes colores para poder identificar cada región.

4. RESULTADOS

El método de segmentación para imagines de rango descrito en este artículo, se desarrollo y ejecutó en un computador Athlon 1.8 con 512 MB de memoria RAM, con treinta imágenes de rango de superficies planas, disponibles en la base de datos de la Universidad del Sur de la Florida, las cuales fueron tomadas con un sensor de rango de luz estructurada ABW [11].

El cálculo de los gradientes se obtuvo con él procedimiento descrito en la Sección 3, utilizando ventanas de 3 x 3, 5 x 5 y 15 x 15, para el cálculo de los gradientes de profundidad, el cálculo de normales para cada píxel y el cálculo de gradientes de orientación, respectivamente. Se tomó un umbral de 15 unidades de rango, para los gradientes de profundidad y de 1 radian para gradientes de orientación.

A cada imagen se le aplicó un filtro de mediana, utilizando una ventana de 9 x 9, antes del cálculo de los gradientes para reducir un poco el ruido presente en las imágenes originales. Los bordes de profundidad definidos por los gradiente de profundidad obtenidos, se adelgazaron con el método de adelgazamiento de Hilditch[2], con lo cual se obtuvieron bordes delgados; pero interrumpidos.

El algoritmo genético presentado en la Sección 3., se implementó con la librería de dominio público GaLib [12], del Instituto Tecnológico de Massachusetts. Se utilizaron las clases GASteadyState para el esquema del algoritmo genético y GA2DBinaryStringGenome para la representación de los individuos. Los pesos asignados a la función de

aptitud del algoritmo genético fueron: WC = 0.5, WPR = 0.3 y WF = 0.2. Estos valores, al igual que los umbrales de profundidad y orientación, fueron seleccionados luego de diversos ensayos, con distintos valores para cada uno de los parámetros.

A partir de los bordes encontrados por el algoritmo genético para cada una de las treinta imágenes de rango de superficies planas de la Universidad del Sur de la Florida, se realizó un rápido etiquetamiento de regiones. Las imágenes etiquetadas obtenidas como resultado se pueden observar en el Anexo 1. Estos resultados fueron evaluados con la metodología propuesta por Hoover, teniendo en cuenta distintos valores de tolerancia: 0.51, 0.60, 0.70, 0.75, 0.80, 0.90 y 0.95. El comportamiento del método propuesto frente a las cinco métricas de la metodología de evaluación (sobresegmentación, subsegmentación, ruido, perdida, detecciones correctas), se puede observar en el Anexo 2.

5. ANALISIS DE LOS RESULTADOS

Durante el desarrollo del método segmentación se puedo comprobar que las treinta imágenes de rango planas de la Universidad del Sur de la Florida utilizadas, contienen muchas imperfecciones en los valores de profundidad y coordenadas de los píxeles. A pesar de haberse aplicado un filtro de mediana para la reducción de ruido, éste persistió, dificultándose el cálculo de los gradientes y por consiguiente, la obtención de los mapas de bordes.

Comparando los mapas de bordes definidos por los gradientes de profundidad y orientación (ej. Figura 1), con los mapas de bordes finales, obtenidos por el algoritmo genético (Anexo 1), se puede observar que se generaron mejores mapas de bordes, delgados, sin fragmentaciones y se desecharon muchos falsos bordes. Sin embargo, no se pudo evitar que los bordes finalmente definidos no conformaran líneas rectas, debido a las mismas imperfecciones de los gradientes calculados.

Las gráficas de desempeño en el Anexo 2., ilustran que al igual que otros métodos de segmentación, que han sido evaluados con la metodología de Hoover [3], se presenta una menor precisión en las detecciones correctas a medida que se aumenta la tolerancia; así mismo, hay una mayor tendencia a incorporar ruidos y pérdidas. Sin embargo, el método de segmentación propuesto tiene como ventaja la definición de bordes no interrumpidos.

6. CONCLUSIONES

En este artículo, se presentó un método de segmentación para imágenes de rango planas, basado en la detección de los bordes de las superficies.

El método de segmentación propuesto está constituido por tres etapas. En la primera etapa se calculan cambios o gradientes de profundidad y orientación entre los píxeles de la imagen; para llevar a cabo esta primera etapa se utilizó el método propuesto por Silva en [9]. Al aplicar el método de Silva se pudo observar que los gradientes calculados se aproximaban mucho a los bordes de las superficies; sin embargo, el conjunto de todos los puntos gradientes daba como resultado, bordes gruesos, interrumpidos y en muchos casos mal situados.

La segunda etapa del método consiste en encontrar un mapa de bordes delgado, sin fragmentaciones, ni falsos bordes, a partir de los gradientes calculados. Esta etapa se lleva a cabo por medio de un algoritmo genético, el cual se encarga de seleccionar puntos gradientes o puntos bordes terminales, en el individuo, para ir definiendo bordes delgados y no fragmentados. En el algoritmo genético diseñado para esta segunda etapa, el operador de mutación, cumple un papel importante, debido a que sobre él recae la tarea de ir seleccionando los puntos necesarios para ir conformando los bordes con las características deseadas.

Una vez, se obtiene un mapa de bordes delgado y sin fragmentaciones, la siguiente etapa consiste en hacer un etiquetamiento de las regiones delimitadas por los bordes. Al etiquetar los píxeles de las regiones definidas por los bordes, no se tuvo en cuenta los píxeles que se encuentran sobre el borde, lo cual pudo afectar la evaluación del método propuesto, utilizando las métricas propuestas por Hoover [3].

El método de segmentación descrito en este artículo se utilizó para segmentar treinta imágenes de rango de superficies planas de la Universidad del Sur de la Florida. Las imágenes segmentadas que se obtuvieron fueron evaluadas con la metodología propuesta por Hoover[3]; los resultados obtenidos, demuestran que el método de segmentación propuesto al igual que otros segmentadores descritos en la literatura, tiene un menor desempeño a medida que se aumenta la tolerancia para la comparación entre la segmentación obtenida y la segmentación manual de una imagen.

Al observar las imágenes segmentadas por el método propuesto, se puede decir que el segmentador desarrollado encuentra las regiones de las imágenes; pero los límites entre ellas no describen muy bien la geometría de los

objetos, esto se debe a que los bordes obtenidos, a pesar de que son delgados y no fragmentados, no son muy rectos. Una dificultad que se tuvo al implementar el método de segmentación descrito, es que las imágenes con las que se trabajó tienen muchas imperfecciones en sus valores de rango, lo cual afectó el cálculo de los gradientes de profundidad y orientación, y el trabajo efectuado por el algoritmo genético.

El desempeño del método de segmentación propuesto puede mejorar notablemente, si se logra realizar un mejor preprocesamiento de las imágenes, con un filtro que elimine las imperfecciones, teniendo en cuenta las propiedades geométricas de las imágenes de rango como son: los valores de las coordenadas 3D, las normales, etc.

Referencias

[1] BESL J, JAIN R, Segmentation through Variable-Order Surface Fitting, IEEE Transactions on PAMI, Vol. 10, number 2, 1988

[2] R. González, Tratamiento Digital de Imágenes, Addison-Wesley, 1996.

[3] A. Hoover, Jean-Baptiste, Goldgof, Bowyer, A Methodology for Evaluating Range Image Segmentation Techniques, *IEEE Workshop on Applications of Computer Vision*, pp. 264-271, Sarasota, FL, December 1994.

[4] Jiang, Bunke, Fast segmentation of range images into planar regions by scan line grouping, Technical report IAM 92-006, Institute for Computer Science, University of Bern, Switzerland, April 1992.

[5] Jiang, Bunke, Edge Detection in Range Images Based on Scan Line Approximation, Computer Vision and Image Understanding, Vol. 73, No. 2, February, pp. 183–199, 1999.

[6] Jiagn, Bowyer, Morioka, Hiura, Sato, Inokuchi, Bock, Guerra, Loke, Dubuf, Some Further Results of Experimental Comparison of Range Image Segmentation Algorithms, 15th Int. Conference on Pattern Recognition, Spain, Sept. 2000.

[7] J. Koza, Genetic Programming II, The MIT Press, 1994, pag. 21-24.

[8] M. Powell, Comparing curved-surface range image segmenters, Master's thesis, Department of Com-puter Science and Engineering, University of Southern Florida, Apr. 1997.

[9] L. Silva, Estudo Sobre Deteção de Bordas em Imágenes de Profundidade, Departamento de Informática UFPR, 2000.

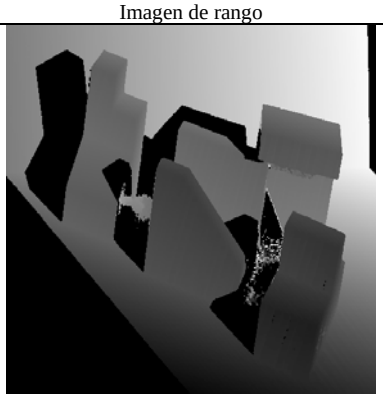
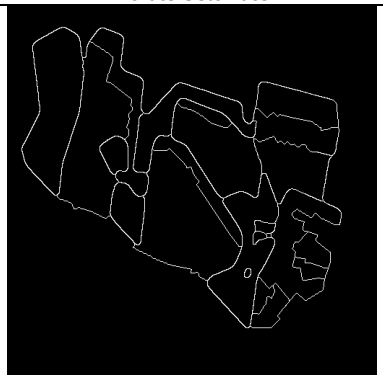
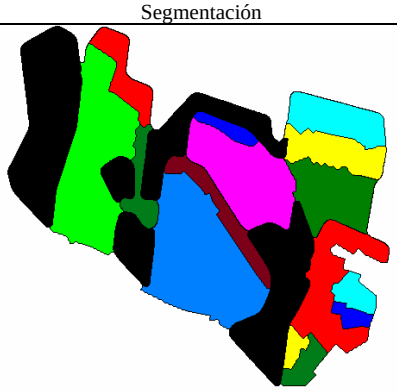
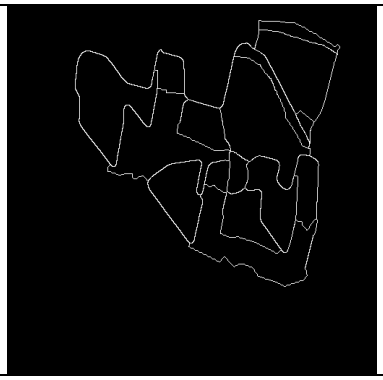
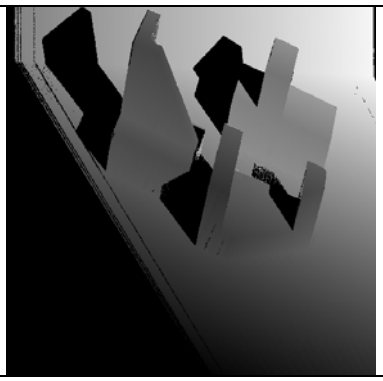
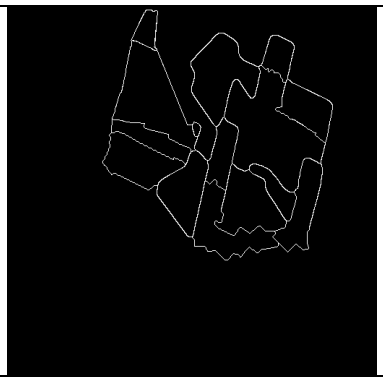
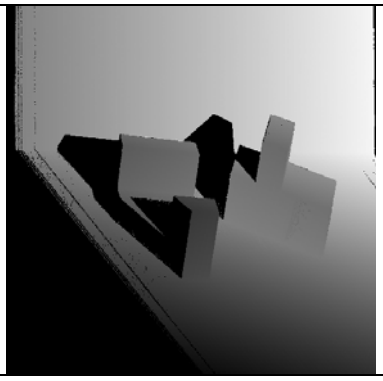
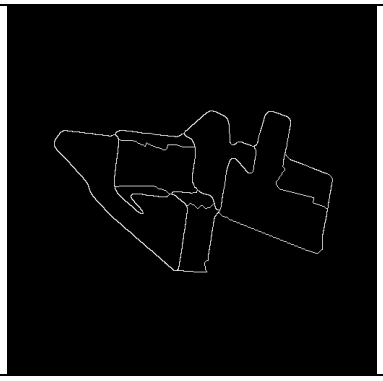
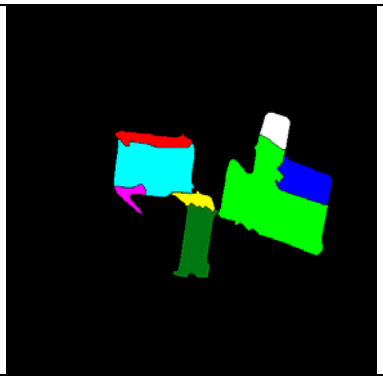
[10] L. Silva, O. Pereira, Segmentação de Imagens de Profundidade por Deteção de bordas, Proceedings of the 21th Brazilian Computer Society Congress, Thesis/Dissertations Contest - 2nd Place Fortaleza/CE - Brazil, 2001.

[11] Range Image Database [online]. The Computer Vision / Image Analysis Research Laboratory at the University of South Florida. Available from internet: < http://marathon.csee.usf.edu/ >

[12] M. Wall, Galib A C++ Library of Genetic Algorithm Components [online]. Massachusetts Institute of Technology, 12 December 1999. Available from internet: <URL: http://lancet.mit.edu/ga/ >.

Anexo 1.

TABLA 1
CUATRO IMAGENES DE RANGO CON SUS RESPECTIVOS MAPAS DE BORDES Y SEGMENTACIONES

Imagen de rango	Bordes Obtenidos	Segmentación
		
		
		
		

Anexo 2.

Graficas de desempeño del segmentador propuesto.

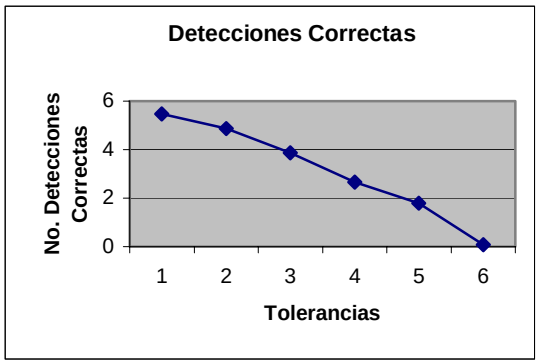


Figura 2. Desempeño frente a las detecciones correctas

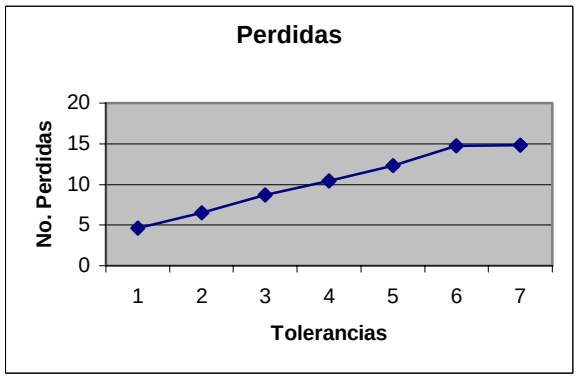


Figura 3. Desempeño frente a las Perdidas

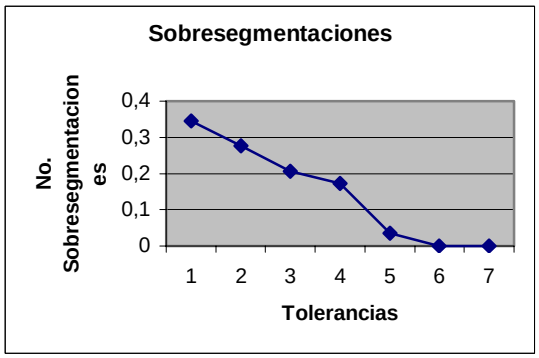


Figura 4. Desempeño frente a las Sobresegmentaciones

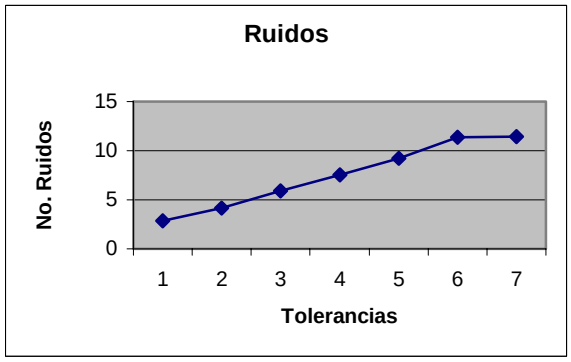


Figura 5. Desempeño frente a los Ruidos

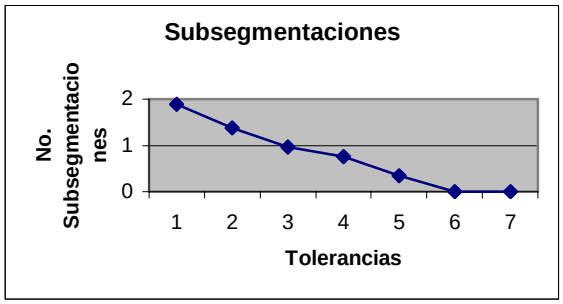


Figura 6. Desempeño frente a las Subsegmentaciones

Myrup: uma Adaptação do RUP para Projetos de Pequeno e Médio Porte

Jocelene de Oliveira Reis
(jo.oliveira@terra.com.br)

Arnaldo Dias Belchior
(belchior@unifor.br)

Mestrado em Informática Aplicada – MIA
Universidade de Fortaleza - UNIFOR
Av Washington Soares, 1321
60.811.341 – Fortaleza – CE – Brasil

Abstract

This paper presents the Software Development Method MyRup. It is a customization of RUP (Rational Unified Process) for media and small projects, applying some CMMI-SW (Capability Maturity Model Integration for Software) and HCI (Human Computer Interface) concepts. This method purposes a sequence between activities in several RUP areas to be a guideline for development and management team. MyRup was successfully applied in some software projects. The learning lessons in these projects were used to adjust the proposed model.

Key-worlds: Development process, process quality, Method of development, RUP.

Resumo

Este trabalho apresenta o método de desenvolvimento de software MyRup, que é uma adaptação do RUP (*Rational Unified Process*) voltado para projetos de pequeno e médio porte, com a utilização de conceitos do CMMI (*Capability Maturity Model Integration para Software*) e IHC (Interface Homem Computador). O método propõe uma seqüenciação entre as atividades a partir das diversas disciplinas do RUP, com a finalidade de guiar efetivamente os desenvolvedores desses projetos, sendo um guia simplificado e de fácil entendimento. O MyRup foi aplicado com êxito em alguns projetos de software, em uma pequena empresa de software. As lições aprendidas nesses projetos estão sendo utilizadas para refinarem o modelo proposto.

Palavras-chave: Processo de Desenvolvimento, qualidade do processo, método de desenvolvimento, RUP.

1 INTRODUÇÃO

Segundo estatísticas recentes, 74 % dos projetos de software não são bem sucedidos [12]. Assim sendo, a indústria de software, incluindo pequenas e médias empresas, buscam cada vez mais alternativas para mudar esta realidade.

Neste contexto, o RUP (*Rational Unified Process*) [2, 7] surgiu como uma alternativa, consolidando a utilização da UML (*Unified Modeling Language*) [1, 6], o conceito de arquitetura de software [10] e uma definição das atividades a serem executadas no desenvolvimento de um projeto de software.

O RUP [7, 11] é um processo com o objetivo de atender a uma grande classe de diversos projetos, podendo ser considerado como um *framework* genérico, necessitando de configurações para ser usado efetivamente. Assim, a sua implantação requer uma série de adaptações para a realidade particular da organização, dificultando a sua utilização em pequenas empresas, pois dificilmente estas têm um grupo de profissionais da área de qualidade aptos a fazerem tais adaptações.

Atuando neste problema, o MyRup propõe-se a ser um método baseado no RUP adicionando conceitos de IHC (Interface Homem Computador), adaptado a realidade de pequenos e médios projetos, oferecendo um guia simplificado, de fácil entendimento e passo a passo sobre as diversas atividades a serem executadas pela equipe de desenvolvimento durante o projeto.

O MyRup, ao invés de apresentar a seqüência das atividades por *disciplina* como o RUP faz, apresenta a seqüência destas atividades por fase, mostrando claramente o encadeamento destas em cada momento do projeto.

Este método utiliza um subconjunto das atividades propostas pelo RUP a fim de tornar o processo mais simples e acrescenta aspectos de usabilidade [9] através do uso do modelo do usuário [3]. O MyRup continua focado em casos de uso [14] e arquitetura [10], porém adiciona a preocupação com interface e de oferecer uma ligação clara entre as atividades das diversas disciplinas em todo o ciclo de desenvolvimento.

O artigo está organizado da seguinte forma: a seção 2 descreve em linha gerais o RUP; a seção 3 apresenta o MyRup com suas principais características e fases; a seção 4 mostra o estudo de caso onde o MyRup foi aplicado e apresenta os resultados preliminares obtidos; a seção 5 exibe as principais conclusões deste trabalho.

2 ENFOQUES SOBRE O RUP

O RUP (Figura 1) é um processo de desenvolvimento de software iterativo, centrado em arquitetura e dirigido por casos de uso. Oferece uma abordagem baseada em disciplinas (*workflows*) para atribuir tarefas e responsabilidades dentro de uma organização de desenvolvimento, tendo como principal meta garantir a produção de software de alta qualidade, atendendo às necessidades dos usuários dentro de um cronograma e orçamento previsíveis [11].

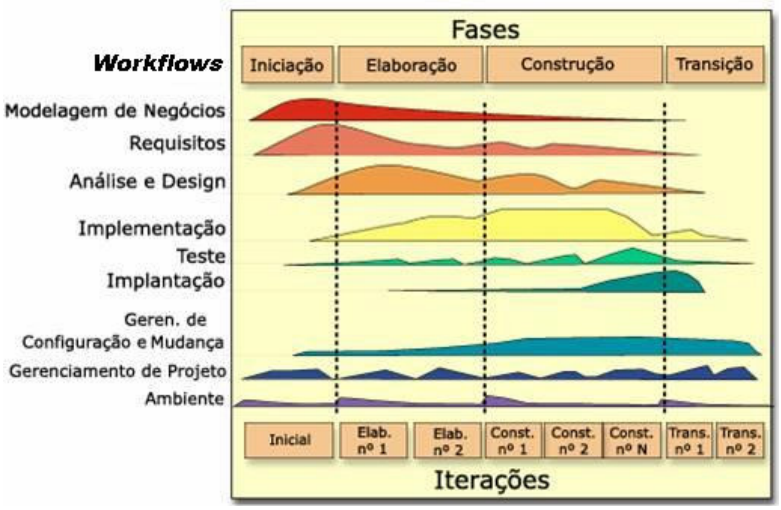


Figura 1: Fases e disciplinas do RUP

O RUP possui duas dimensões: horizontal e vertical. O eixo horizontal representa o aspecto dinâmico do processo, mostrando o tempo e os aspectos do ciclo de vida à medida que o projeto se desenvolve. O eixo vertical representa o aspecto estático do processo, mostrando as disciplinas que agrupam as atividades a serem desenvolvidas.

Uma disciplina envolve todas as atividades a serem realizadas para produzir um determinado conjunto de artefatos. O principal objetivo do agrupamento de atividades em disciplinas é ajudar a compreender o projeto a partir de uma perspectiva em cascata 'tradicional', como, por exemplo, ver todas as atividades a serem desenvolvidas de

requisitos. A separação dessas atividades em disciplinas distintas pode facilitar a compreensão, mas dificulta a programação, pois a seqüência das atividades do projeto como um todo não é clara.

O RUP divide o projeto de software em 4 fases distintas e com objetivos a serem alcançados: *concepção*, *elaboração*, *construção*, e *transição*.

A fase de concepção tem como objetivo fornecer uma visão inicial do projeto, definindo o escopo além de estimar preliminarmente os riscos e custos do projeto. A fase de elaboração tem por objetivo analisar o domínio do problema, de modo a propor uma arquitetura para os cenários da aplicação. O objetivo da fase de construção é esclarecer os requisitos restantes e concluir o desenvolvimento do sistema com base na arquitetura definida.

Finalmente, a fase de transição tem por objetivo assegurar que o software seja disponibilizado para seus usuários finais. A fase de transição inclui testar o produto e realizar pequenos ajustes com base no *feedback* do usuário. Ao final da fase de transição, os objetivos devem ter sido atendidos e o projeto deve estar em uma posição para fechamento.

A seguir, será apresentada a proposta do MyRup, que é um modelo de desenvolvimento de software para pequenos e médios projetos, baseado no RUP e em conceitos de IHC [9].

3 O MyRUP

O MyRup visa ser um guia simples e passo a passo para os integrantes da equipe de desenvolvimento de software, preservando a essência do RUP, porém trocando a forma de visualização das atividades (no RUP por disciplinas; no MyRup por fases), e acrescentando atividades com o enfoque de garantir a usabilidade do software. Assim, o método mostra uma forma de adaptação do RUP para pequenos projetos, já que o RUP é um framework genérico.

A característica essencial do MyRup e diferença em relação ao RUP é o agrupamento de atividades das diversas disciplinas por fase, mostrando uma visualização seqüencial dos grupos de atividades e tarefas a serem executadas em cada fase do processo (concepção, elaboração, construção e transição). Essa forma de visualização foi proposta para tirar a dificuldade existente no RUP de saber a ordem das atividades a serem executadas durante o projeto.

No RUP, a seqüência das atividades é mostrada por *disciplina* (exemplo: seqüência das atividades de requisitos do projeto e a seqüência de atividades de planejamento), não deixando clara a ligação das atividades das diferentes disciplinas, por exemplo, se a atividade de fazer o cronograma vem antes ou depois da atividade de especificação de requisitos.

Já a visualização do MyRup permite que o desenvolvedor possa conhecer a seqüência das atividades a serem executadas em cada fase do projeto, abstraindo-se da *disciplina* a que esta pertence. O desenvolvedor irá ver apenas a ordem que essas tarefas devem ser desempenhadas no decorrer do projeto (Ex: Especificar requisitos e depois fazer o cronograma).

O MyRup retirou a disciplina de *modelagem de negócio* e várias atividades de outras disciplinas para tornar o processo mais leve para pequenos e médios projetos. Esse subconjunto visou ter o mínimo de atividades do RUP para um projeto garantir a qualidade de desenvolvimento. A outra diferença fundamental do método é o uso do modelo do usuário durante várias atividades ao longo do ciclo de vida, com o objetivo de adicionar aspectos de usabilidade na construção do software.

A Figura 2 apresenta o fluxo principal do MyRUP para o desenvolvimento de um projeto, com seus principais fluxos de trabalho, como se o ciclo de vida adotado para o projeto fosse em cascata. Cada fluxo de trabalho é detalhado em atividades envolvendo atores e gerando artefatos, além da exemplificação de artefatos.

Como indicado na Figura 2, o método procura guiar a equipe de desenvolvimento desde o início do projeto, com o fluxo *Conceber Projeto* (este contem as atividades de negociação com o cliente e de confecção de uma proposta), passando pela definição do sistema e planejamento macro.

Depois desta fase inicial, é proposta a definição e o refinamento da arquitetura a ser adotada (fluxo *Definir Arquitetura*), para só depois a implementação começar, seguida pelos testes e a implantação da versão final. Em conjunto com estes fluxos, ocorrem as atividades de gerenciamento do projeto com as atividades de planejamento e acompanhamento de cada iteração, além de controle dos riscos, gerência dos requisitos (com a finalidade de defini-los e controlar suas possíveis mudanças), e o gerenciamento da configuração dos artefatos dos projetos.

O MyRup indica que o projeto adote o ciclo de vida interativo, para minimização dos riscos e maiores chances de sucesso [13].

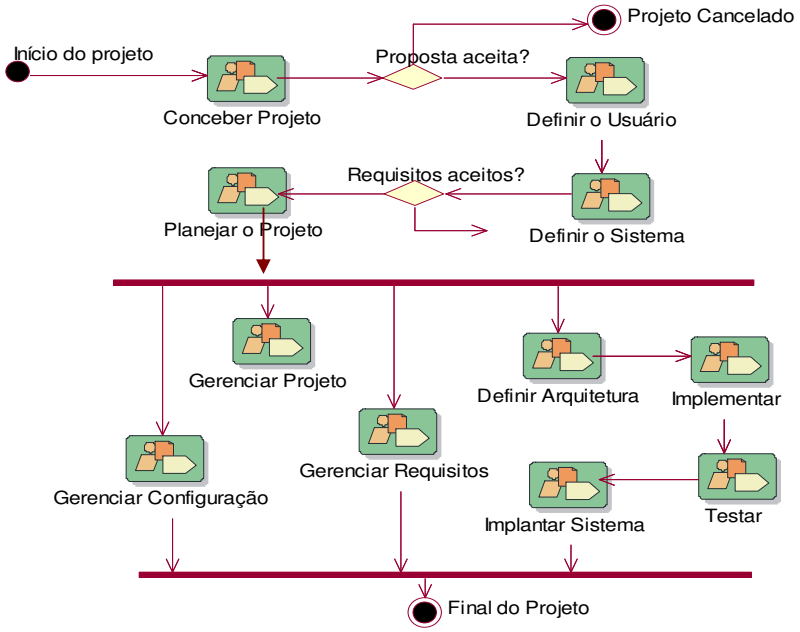


Figura 2: Fluxo principal do MyRup

Segue-se a apresentação de cada uma das fases do MyRup: *concepção*, *elaboração*, *construção*, e *transição*.

3.1. Concepção

A *Fase de Concepção* é a primeira fase do MyRup e tem como principal objetivo alcançar o entendimento entre todas as partes envolvidas (cliente, usuários, gerentes e equipe de desenvolvimento) sobre o produto a ser desenvolvido. Nessa fase, o escopo, os objetivos do sistema, o grau de usabilidade desejado e os principais marcos devem ser bem definidos e entendidos por todos. A fase de concepção é bastante crítica para todos os projetos, pois segundo Tom De Marco [5], 56% dos erros de um software podem ser rastreados na fase de requisitos.

Nas atividades do fluxo de levantamento das necessidades dos usuários, foram incluídas atividades de modelagem do usuário, para que a equipe de desenvolvimento conheça o perfil destes e suas características, a fim de conseguir levantar os requisitos com maior facilidade além de influenciar no desenvolvimento do software.

A Figura 3 mostra os grupos de atividades (fluxos de trabalho) a serem executados dentro da fase de concepção, enfatizando a ordem em que estas ocorrerão. Com esta ordenação, o método tenta mostrar aos desenvolvedores tudo o que devem fazer na iteração da fase de Concepção.

Um exemplo de abstração das atividades das disciplinas é o fluxo *Planejar o desenvolvimento do Projeto* (pertencente à *disciplina* Gerenciar Projeto), ocorrendo depois do fluxo *Definir Sistema* (pertencente à *disciplina* Gerenciamento de requisitos). A equipe deve abstrai-se dos detalhes de organização do modelo, atendo-se apenas que precisa inicialmente definir o sistema, e depois planejar seu desenvolvimento.

No caso da fase de Concepção ter mais de uma iteração, o fluxo dessa fase repetir-se-á para cada uma delas. No contexto de pequenos e médios projetos, o MyRup recomenda que haja apenas uma iteração nessa fase.

Cada fluxo de trabalho apresentado na Figura 3 é descrito e diagramado, mostrando as atividades que os compõem, os artefatos gerados e os papéis desempenhados. Para exemplificar como o método é estruturado, será exibido a seguir um resumo do detalhamento de um fluxo e a descrição de uma de suas atividades, assim como uma parte de um dos artefatos a ser gerado.

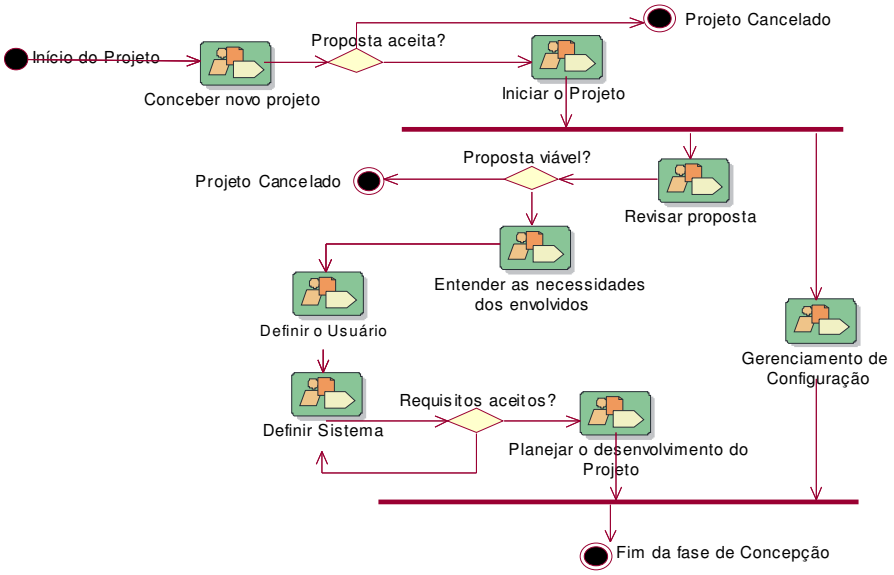


Figura 3: Fase de Concepção do MyRup

Na Figura 4, é apresentado um resumo do detalhamento do fluxo de trabalho *Entender as necessidades dos envolvidos* com suas atividades e o papel a ser desempenhado pelos integrantes da equipe de desenvolvimento. Esse fluxo apresenta as atividades de levantamento das informações com clientes ou usuários, para se obter a compreensão das reais necessidades do produto a ser desenvolvido pelo projeto (*Entrevistar os usuários*).

Inicialmente, pode-se imaginar essa atividade como um levantamento das expectativas de usuários e clientes, que poderão ser concretizadas em requisitos. A atividade *Capturar características dos usuários* visa levantar o perfil dos usuários assim como sua experiência no domínio do problema, para poder guiar desde o levantamento dos requisitos até a construção de uma interface com alto ou baixo grau de usabilidade.

Esse fluxo visa também *Capturar o vocabulário do cliente*, como um meio de uniformizar a linguagem que será utilizada pela equipe de desenvolvimento, pelos clientes e usuários.

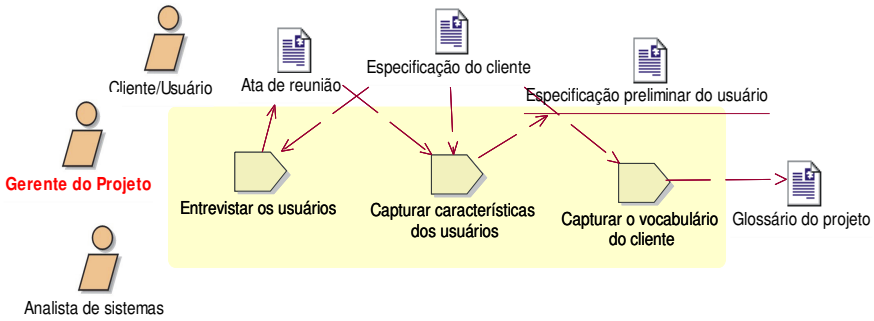


Figura 4: Fluxo *Entender as necessidades dos envolvidos*

A equipe do projeto, através de entrevistas com o cliente, ou em conversas técnicas internas, captura o vocabulário e elabora um glossário do projeto (artefato) com os termos e as definições necessárias. A definição do glossário é de suma importância para a consistência dos documentos ao longo do projeto (ex: especificação de requisitos, documento de arquitetura, etc.), a fácil inserção de novos membros à equipe, e o entendimento de documentos técnicos pelo cliente e de documentos de negócios pela equipe do projeto.

3.2. Elaboração

O MyRup propõe uma *Fase de Elaboração* mais simplificada que a do RUP, basicamente com atividades de *definir arquitetura* gerando o artefato Especificação de arquitetura e seu refinamento (fluxo "*refinar arquitetura")* através dos diagramas de classe e de seqüência. A idéia dessa fase é diminuir os riscos da implementação do sistema através da definição de uma arquitetura sólida e já testada em protótipos evolutivos nas iterações. A Figura 5 mostra a visão geral da Fase de Elaboração do MyRup.

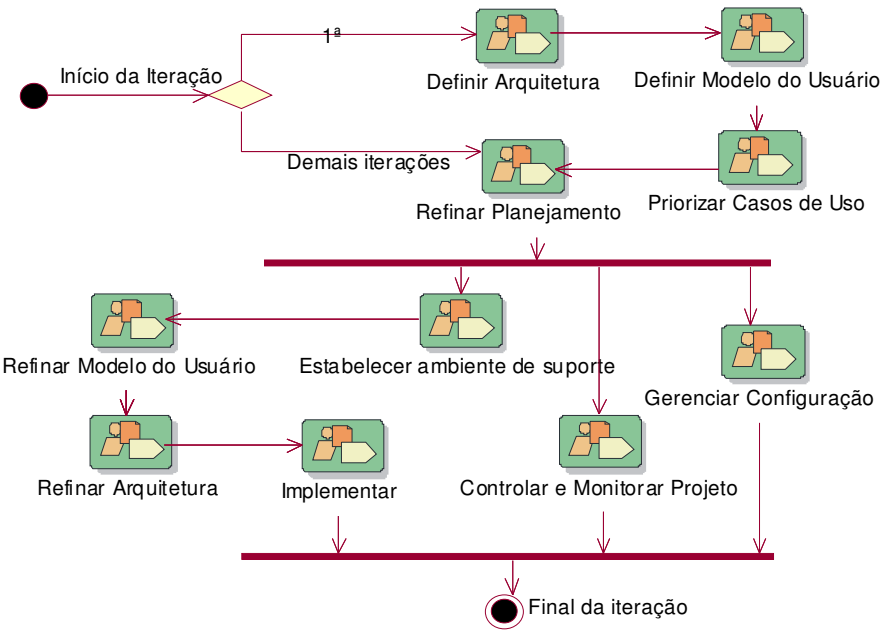


Figura 5: Fase de Elaboração do MyRup

Os principais fluxos de trabalho dessa fase são: *Definir arquitetura* e *Refinar a arquitetura* além de *Definir usuário*. O fluxo *Definir arquitetura* leva a equipe a estabelecer como o sistema irá funcionar em termos de componentes e tecnologias. A idéia nesse fluxo é de estabelecer uma arquitetura macro válida para todo o projeto visando identificar desde cedo as suas limitações e implicações (ex ao definir um banco de dados que o não suporta *store procedure*).

Após a definição da arquitetura, a equipe escolherá os casos de uso relevantes, no fluxo *Priorizar casos de uso*, para validar a arquitetura proposta. Esses casos de uso normalmente são considerados o núcleo do sistema, ou seja, o subconjunto capaz de validar se a arquitetura proposta irá funcionar.

O fluxo *Definir usuário* modela os perfis mais importantes do usuário, com suas diferentes expectativas do sistema em termos de funcionalidades e usabilidade. Essa definição acarretará num cuidado especial para funcionalidades ou interfaces que o usuário acha relevante, tanto por não conhecer ou por achar muito relevante.

O fluxo *Refinar arquitetura* (Figura 6) é um dos fluxos de maior valor percebido pelos implementadores, pois faz a interligação da documentação feita nos casos de uso na fase preliminar com o código a ser implementado. O MyRUP propõe apenas o uso dos diagramas de classe, de seqüência, e de estado (quando necessário para guiar o desenvolvimento).

Nestas atividades, o método mostra como chegar dos casos de uso ao código. Os desenvolvedores ou projetistas (*designers*) vão detalhar cada caso de uso (fazer a descrição destes e seus fluxos básico e alternativos), realizá-los através dos diagramas de classe e de seqüência e, quando a ferramenta o permitir, gerar o código a partir dos próprios diagramas desenvolvidos.

Por meio desta estrutura, objetiva-se mostrar aos desenvolvedores que a documentação é útil e está sendo feita como passo para a geração muitas vezes automática e com qualidade do código.

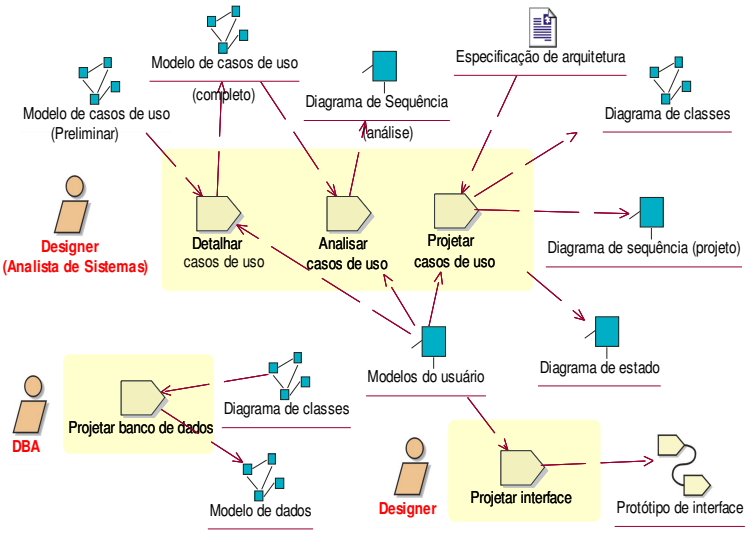


Figura 6: Detalhamento do Fluxo *Refinar Arquitetura*

O fluxo Implementar enfoca a geração do código dos casos de uso priorizados a fim de validar a arquitetura proposta. Normalmente essa versão não é entregue ao cliente, porém gerando um protótipo evolutivo para os demais incrementos.

3.3. Construção

No MyRup, a *Fase de Construção* é voltada para um planejamento baseado em casos de uso e nas atividades voltadas para a implementação de uma versão passível de ser instalada no cliente. É interessante que o gerente do projeto planeje várias iterações para a fase de construção, e cada uma destas seguirá o fluxo apresentado na Figura 7.

Com a finalidade de preparar uma versão possível de ser instalada no cliente, aparece pela primeira vez, um fluxo de trabalho especifico para testes a fim de garantir a qualidade de uma versão.

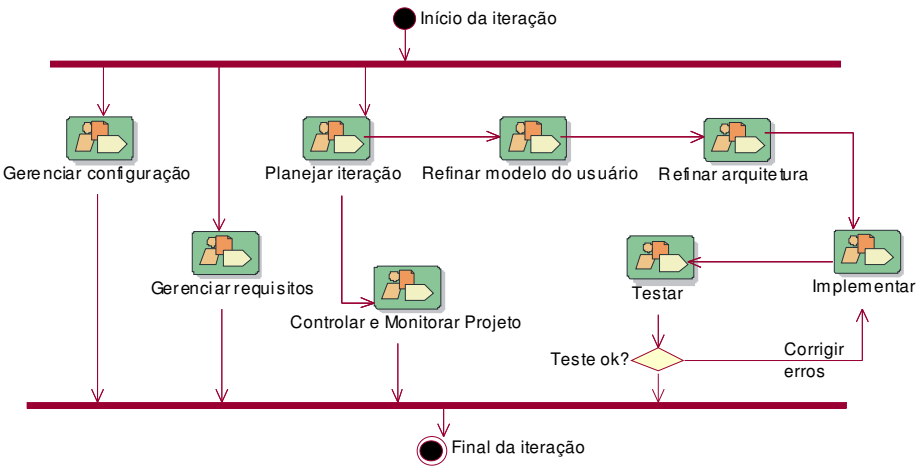


Figura 7: Fase de Construção do MyRup

A Figura 8 apresenta as atividades do fluxo *Planejar iteração* da fase de construção.

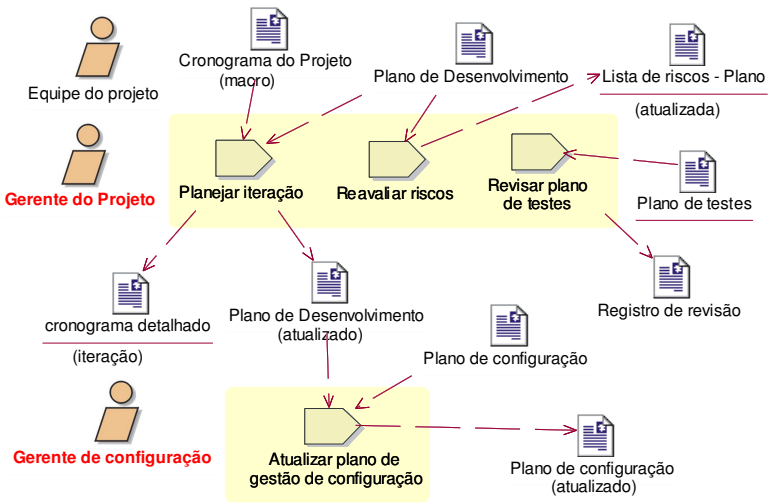


Figura 8: Detalhamento do Fluxo *Planejar iteração*

No fluxo *Planejar iteração*, o Plano de desenvolvimento, que foi planejado a partir de conceitos do CMMI [4], contém os casos de uso de cada iteração, enquanto que o cronograma macro possui os marcos negociados com o cliente.

Com estas informações, o gerente do projeto organiza o cronograma detalhado da iteração, contendo todas as atividades a serem desenvolvidas. O cronograma proposto pelo MyRUP é guiado por casos de uso, conforme a Tabela 1 (parte do artefato gerado na seção 4).

Tabela 1: Exemplo de artefato - Cronograma detalhado do projeto (parcial)

Tarefas	Nome da tarefa	Duração (dias)	Data Início	Responsável
	PROJETO SISC			
1	**Fase de Construção – Iteração 1**		01/04/04	
2	Reunião de Planejamento	1	02/04/04	Todos
3	Atualizar Plano de Desenvolvimento	1	08/04/04	Sellaro
	…. (outras atividades)	1	09/07/04	Ariluce e Fábio
6	***Detalhar Casos de Uso***			
7	Criar Fatura	1	10/04/04	Fábio
8	Inserir transações	0,5	11/04/04	Fábio
	... (demais casos de uso da iteração)			
16	***Projetar Casos de Uso***			
17	Criar Fatura	1,5	14/04/04	Ariluce
18	Inserir transações	2	16/04/04	Ariluce
	... (demais casos de uso da iteração)	2	16/04/04	Fábio
	Implementar Caso de Uso			
17	Criar Fatura	1,5	14/04/04	Ariluce
	... (demais casos de uso da iteração)			
21	***Testar Caso de Uso***	4	10/04/04	Sellaro
23	Criar Fatura	2	14/04/04	Fábio
	... (demais casos de uso da iteração)			

O fluxo *Refinar arquitetura* da fase de construção é semelhante ao da fase de elaboração com a diferença de que os diagramas de analise tornam-se opcionais, pois normalmente não são desenvolvidos devido à experiência que a equipe já adquiriu em confeccionar diagramas, e pelo conhecimento adquirido no domínio do problema.

O refinamento é feito apenas para o grupo de casos de uso planejados para a iteração corrente. A vantagem da implementação e modelagem em incrementos é o amadurecimento da equipe na técnica de confeccionar esses diagramas e no domínio do projeto, minimizando a quantidade de erros no projeto do software.

A Figura 9 apresenta um diagrama de classe construído em um dos projetos do estudo de caso (seção 4), que ilustra as classes utilizadas na realização de um dos casos de uso, e que já estão prontas para a geração de código.

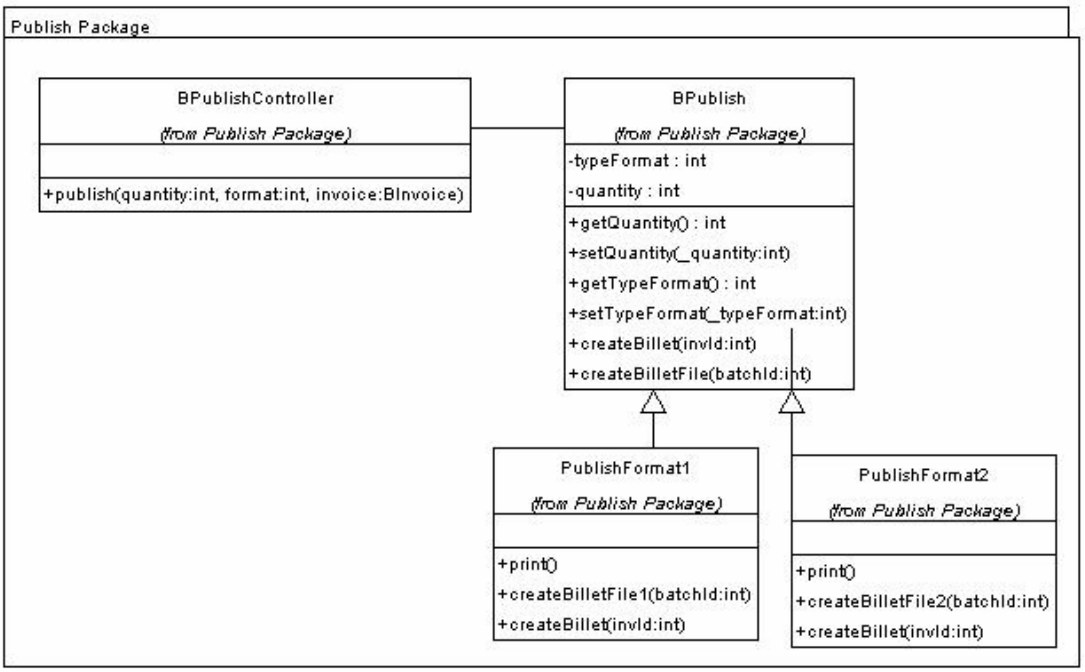


Figura 9: Exemplo de artefato - Diagrama de Classes do projeto (parcial)

As atividades do fluxo *Gerenciar requisitos* nesta fase assumem uma importância bastante relevante, pois uma mudança em algum requisito, pode impactar desde uma simples mudança em um método de uma classe até uma alteração na arquitetura do sistema ou escopo do projeto.

Já o resultado da modelagem do usuário influencia fortemente na implementação das interfaces, pois estas devem ser projetadas de acordo com os perfis e conhecimentos dos usuários. A experiência do usuário e o tipo de componentes gráficos preferidos guiam o desenvolvimento para a construção do software que o usuário goste de usar e atenda suas expectativas de usabilidade.

3.4. Transição

A *Fase de Transição* do MyRup (Figura 10) é voltada para a implantação da versão final no cliente. Portanto, o gerente do projeto só deve dar por concluídas todas as iterações da Fase de Construção e passar para a primeira iteração da Fase de Transição, quando a implementação já tiver sido concluída.

O enfoque da Fase de Transição é realizar todas as atividades necessárias para a implantação e finalização do projeto, desde a documentação final do sistema ao teste de integração e treinamentos com os usuários. Depois da última iteração dessa fase, tem-se a conclusão do projeto.

A partir deste momento, as futuras manutenções devem ser tratadas como mini-projetos, com a elaboração de seu planejamento, e passando por todas as fases do MyRup.

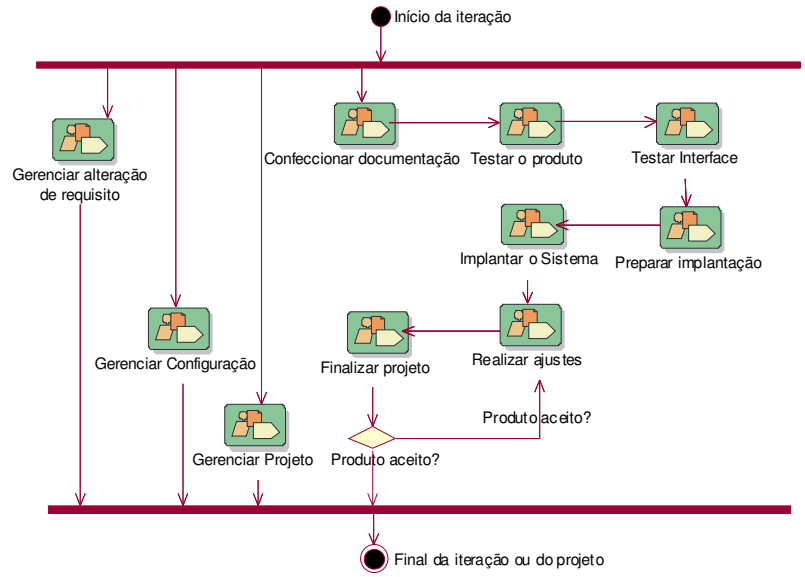


Figura 10: Fase de Transição do MyRup

O gerente do projeto deve ter um cuidado especial com as atividades do fluxo *Preparar implantação*, (Figura 11), contendo as atividades *Implementar correções*, *Criar arquivos de configuração* e *Fazer empacotamento do produto*, pois freqüentemente essas atividades não são tratadas devidamente pela equipe de desenvolvimento. Porém, uma falha nessas atividades pode gerar transtorno na implantação e, como conseqüência, deixar uma impressão ruim nos clientes e usuários do novo produto de software.

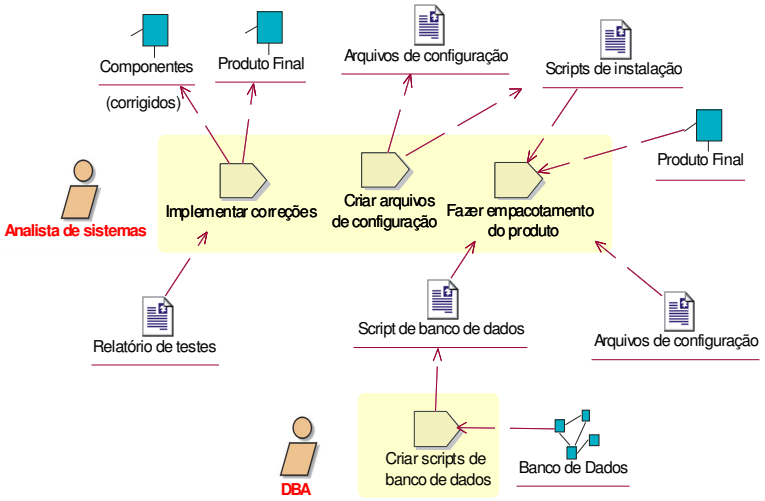


Figura 11: Detalhamento do Fluxo Preparar implantação

O fluxo *Gerenciar configuração* também é extremamente importante na fase de transição, visto que os erros mais freqüentes na implantação estão as versões erradas disponibilizadas aos clientes e usuários. Neste sentido, as *baselines* de *patch* devem ser rigorosamente controladas, para que o cliente e a equipe de implantação possam saber quais os erros encontrados naquela versão, e que correções já foram efetuadas.

O fluxo *Confeccionar Documentação* tem uma grande influência da modelagem dos usuários para gerar uma documentação fácil e de acordo com as características de cada perfil identificado. Como exemplo tem-se o Manual do Usuário desenvolvido de acordo com as premissas de cada perfil do usuário do sistema.

A conclusão do projeto é marcada pelo conjunto de atividades do fluxo *Finalizar projeto*, que, além de tratar da finalização interna do projeto, procura também a obtenção de um aceite formal de término do projeto emitido pelo cliente.

4 ESTUDO DE CASO

O MyRup está sendo aplicado na Fujitec, empresa de TI voltada para desenvolvimento de sistemas embutidos, soluções com *smartcards* e transações eletrônicas, com sede em Fortaleza (Brasil).

A Fujitec é certificada ISO 9001 (BVQI) e, atualmente, está implantando o MyRup, visando obter a qualidade necessária no processo de desenvolvimento de software, para continuar atuando competitivamente no mercado internacional. Atualmente, o MyRup está sendo utilizado em 3 projetos pilotos, visando o refinamento do mesmo e, posteriormente, ser estendido para toda a empresa.

Durante a implantação do MyRup nos projetos pilotos, foram colhidas algumas informações sobre o resultado da implantação do método, apesar de nenhum deles ainda ter sido finalizado para a coleta e a analise dos dados consolidados. A seguir, são apresentados alguns resultados mais relevantes obtidos com o uso do MyRup nesses projetos.

- **Interfaces adaptadas a diversos usuários**
  - A equipe de desenvolvimento conseguiu gerar interfaces adaptadas às necessidades de mais de um grupo a partir da modelagem do usuário. Nessa modelagem os desenvolvedores mapearam as principais exigências e preferências dos usuários como por exemplo a utilização de interfaces extremamente simples e diretas no caso das movimentações nos terminais de venda (POS).
  - Uma segunda conseqüência dessa modelagem e não prevista inicialmente como ganho pelo método (relato das equipes), foi a análise do impacto das decisões de interface no projeto e na construção do software.
- **Facilidade em seguir o processo**
  - Os desenvolvedores conseguiram seguir o MyRup sem uma equipe de qualidade na empresa e nem uma consultoria periódica. Os desenvolvedores depois de participaram de workshops (1 de apresentação e outro prático de estudo de caso) e terem acesso ao processo acharam-se aptos a seguirem as atividades propostas sem um suporte especializado.
- **Conhecimento das tarefas do projeto**
  - A equipe conseguiu visualizar nos fluxos de trabalho propostos e no seu detalhamento a seqüência das tarefas a serem realizadas assim como sua ordem.
  - A experiência com a implantação do MyRup indica que os desenvolvedores não sentiram necessidade de saber a qual o *workflow* pertencia cada atividade (característica do RUP perdida com a adaptação proposta pelo método) e sim sua ordem e o entendimento desta.
- **Melhoria da documentação e a utilização da mesma durante todo o projeto**
  - A Tabela 2 mostra a documentação dos projetos antes e depois do MyRup. A equipe de desenvolvimento atribuiu a efetiva utilização da documentação pelo fato de um documento estar relacionado a outro, pois, para confeccionar um artefato são utilizados como insumos artefatos produzidos anteriormente.
  - Clientes destes projetos reconheceram explicitamente a organização da empresa, e elogiaram, por exemplo, a qualidade da documentação da especificação de requisitos de seu projeto.

Tabela 2: Comparação da documentação antes e depois da implantação do MyRup

Documentação dos projetos antes do MyRup	**Documentação atual dos projetos (Principais documentos)**	
▪ Documento de arquitetura ▪ Modelo de Entidade e Relacionamento ▪ Cronograma macro ▪ Manual do usuário	▪ Especificação de Requisitos ▪ Diagramas de classe ▪ Modelo de Entidade e Relacionamento ▪ Documento de arquitetura ▪ Manual do usuário	▪ Modelo de casos de uso ▪ Diagramas de seqüência ▪ Plano de Desenvolvimento ▪ Cronograma detalhado para cada iteração ▪ ... outros artefatos do método

- **Orientação a Objetos**
  - Com a utilização do MyRup, os projetos passam a trabalhar no paradigma orientado a objeto, usando a UML [8].

- **Facilidade na inserção de novos membros nas equipes de projeto**
  - Os gerentes de projeto constataram que a inserção de novos integrantes nas equipes dos dois projetos pilotos foi bem mais fácil em relação aos demais projetos da empresa. Isto porque logo na primeira semana, o novo integrante da equipe estudava todos os artefatos a partir da documentação do projeto gerados pelo MyRup, e tirava suas duvidas com os demais membros da equipe. Em pouco tempo, o recém membro já se encontrava apto para executar suas tarefas no projeto.
  - O indicador de tempo para que um novo membro começasse a ser produtivo (desenvolver atividades sozinho) diminuiu de três para apenas uma semana.
- **Maior controle no gerenciamento do projeto e estabelecimento de estimativas mais realistas**
  - A utilização do cronograma detalhado por caso de uso fez com que os gerentes acompanhassem melhor as atividades de cada um dos envolvidos no projeto e conseguissem estabelecer estimativas mais próximas da realidade.
- **Controle das versões liberadas para os clientes**
  - Os dois projetos começaram a trabalhar com gestão de configuração e ter um controle mais efetivo nas versões internas e liberadas para o cliente. Para a gestão de configuração, a empresa adotou a ferramenta CVS (*Concurrent Versions System*), que uma ferramenta de software livre, largamente utilizada.

5 CONCLUSÕES

A principal motivação para o trabalho foi propor uma adaptação do RUP voltada para a realidade de pequenos e médios projetos, acrescentando aspectos de usabilidade e uma visualização mais clara da seqüência das atividades. O MyRUP selecionou um subconjunto das atividades do RUP com a finalidade de tornar o processo mais leve e possível de ser seguido com poucas pessoas.

A visualização por fase das diversas atividades dos *workflows* surgiu pela dificuldade em se saber a seqüência das atividades no RUP durante o projeto. Já as atividades de usabilidade foram acrescentadas ao método pelo fato de o RUP não possuir essa característica e ser um fator importante para o desenvolvimento de um produto com qualidade, principalmente na ótica do cliente.

Durante a implantação do método, os membros da equipe fornecem feedback sobre a experiência de utilizar o MyRup, sugestões de melhorias e dificuldades encontradas. Essas informações são utilizadas para a geração de novas versões e workshops para resolver dificuldades e problemas de interpretação.

Vale ressaltar que o método ainda está sendo melhorado e para as próximas versões já estão previstas maiores detalhamentos das atividades de configuração e continuidade com as atividades relacionadas à usabilidade.

Como trabalho futuro, vislumbra-se o desenvolvimento de uma ferramenta para apoiar o gerenciamento do projeto que utiliza o MyRUP.

Referências

[1] Booch, G. *The Unified Modeling Language User Guide.* Addilson-Wesley, 1999.

[2] Booch, G., Jacobson, I., Rumbaugh, J. *The Unified Software Development Process.* Addison-Wesley, 1999.

[3] Brusilovsky, P. *User Modeling and User Adapted Interaction*. 1996.

[4] CMMI Product Team. *CMMI for Systems Engineering/Software Engineering*, Version 1.1 Staged Representation (CMU/SEI-2002-TR-029, ESC-TR-2002-029). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, August, 2002.

[5] Demarco, *T. Princípios e Conceitos de Engenharia de Software.* Compucenter, 1996.

[6] Jacobson, I. *Object-Oriented Software Engineering: A Use Case Driven Approach.* Addison-Wesley, 1994.

[7] Kruchten, P., Kroll, P. *The Rational Unified Process Made Easy*. Addison-Wesley. 2003.

[8] Lairman, G. *Utilizando UML e padrões*. Bookman, 2001.

[9] Mayhew, J. Deohah. *The Usability Engineering Lifecycle*. Morgan Kaufmann. 2001.

[10] Mendes, A. *Arquitetura de Software - Desenvolvimento Baseado na Arquitetura.* Editora Campus. 2002.

[11] Rational Software Corporation. *Rational Unified Process*. Version 2001.03.00. CD-ROM, Rational *Software*, Cupertino, California, 2001.

[12] Standish Group. *CHAOS Report*. 2000.

[13] Summerville, I. *Software Engineering*. 1, Addison-Wesley, 1996.

[14] Winters, J., Jacobsonm I. *Applying Use Cases*. Addilson-Wesley. 1998.

Uma Nova Sinalização GMPLS Aplicada às Redes OBS

Fábio Y. Nagahama
Universidade Federal do Pará (UFPA), Programa de Pós-Graduação em Engenharia Elétrica,
Belém, Brasil, 66075-110
nagahama@yahoo.com

Rafael P. Esteves
Universidade Federal do Pará (UFPA), Departamento de Infomática,
Belém, Brasil, 66075-110
rpesteves@ig.com.br

Antônio J. G. Abelém
Universidade Federal do Pará (UFPA), Departamento de Infomática,
Belém, Brasil, 66075-110
abelem@ufpa.br

Michael A. Stanton
Universidade Federal Fluminense (UFF), Instituto de Computação
michael@ic.uff.br

Resumo

Com os recentes aperfeiçoamentos na tecnologia de transmissão óptica, principalmente no que diz respeito à multiplexação por divisão de comprimento de onda ("Wavelength Division Multiplexing-WDM") viu-se o surgimento de pesquisas no entuito de transpor os limites impostos pela comutação eletrônica. Dentro do contexto da integração IP/GMPLS sobre WDM o paradigma OBS ("Optical Burst Switching") vem gradativamente recebendo maior atenção, pois possui características que lhe proporcionam várias vantagens em relação aos outros paradigmas de comutação óptica. Contudo, para que se possa melhor aproveitar as características do processo de sinalização sem confirmação do paradigma OBS é necessário que os mecanismos de sinalização e controle do GMPLS sejam modificados. Este artigo apresenta sugestões de alterações nos mecanismos de sinalização e controle do GMPLS para melhor adequá-lo ao contexto da comutação de rajadas ópticas rotuladas. Os impactos dessas mudanças foram avaliados através de simulações no ambiente de simulação NS ("Network Simulator").
Palavras chaves: Redes de Alta Velocidade, IP sobre WDM, Comutação de rajadas ópticas (OBS), MPLS Generalizado (GMPLS).

Abstract

With the recent improvements in optical transmission technology, especially after the invention of the wavelength division multiplexing (WDM), several researches has appeared proposing ways to surpass the limits imposed by electronic comuting. Within the context of IP/GMPLS over WDM integration the OBS (Optical Burst Switching) paradigm is gradually gaining greater attention, as it offers a number of advantages when compared with others optical switching paradigms. However, it will be necessary to alter the signalling and control mechanisms of GMPLS, if one wishes to take most effective advantage of unconfirmed signalling as used in OBS. This article presents a number of suggestions for altering the signalling and control mechanisms used in GMPLS, to make them more suitable for use with labelled optical burst switching (LOBS). In order to evaluate the applicability of these proposals, we carry out simulation studies, using the Network Simulator (NS) platform.
Keywords: High-speed networks, IP over WDM, optical burst switching (OBS), Generalised MPLS (GMPLS).

1. Introdução

As novas perspectivas de pesquisa que estão surgindo da combinação das tecnologias IP com WDM ("Wavelength Division Multiplexing") apresentam uma excelente oportunidade para reformular alguns aspectos dos mecanismos de sinalização e controle das redes ópticas, para melhor adequá-los às futuras gerações de inter-redes puramente ópticas. Dentro deste contexto evolutivo, as tecnologias OBS ("OBS – Optical Burst Switching") [QIA99] e GMPLS ("Generalized Multiprotocol Label Switching") [MAN03] vêm se destacando por sinalizarem com soluções mais flexíveis e eficientes na alocação de recursos.

O paradigma OBS vem gradativamente recebendo maior atenção, pois, quando comparado com a comutação de lambdas (comutação de circuitos no contexto óptico) e a comutação de pacotes ópticos ("OPS – Optical Packet Switching") [GUI98], possui características que lhe proporcionam diversas vantagens [ABE02]. No paradigma OBS, diferentemente da comutação de lambdas, os canais são alocados dinamicamente, de acordo com a demanda de transmissão. O processo de reserva do canal é rápido, simples e feito sem confirmação, através do envio de um pacote de controle ("BCP – Burst Control Packet"). O BCP é processado eletronicamente por todos os nós intermediários e transita fora da banda, em um canal de sinalização separado. As rajadas de dados (grupo de pacotes) de comprimento variável são mantidas no domínio óptico ao longo dos nós intermediários e enviadas em seguida ao envio do BCP. Caso o BCP falhe no processo de reserva dos recursos em um nó intermediário qualquer, a rajada será descartada.

O GMPLS estende a arquitetura MPLS (Multiprotocol Label Switching) para atender também a dispositivos nos quais o plano de encaminhamento não reconheça nem limites de pacotes, nem de células, e, em função disso, não sejam capazes de encaminhar dados baseados em informações transportadas tanto no cabeçalho de pacotes como no de células. Especificamente, tais dispositivos fazem parte de tecnologias onde a decisão de encaminhamento é baseada em fatias de tempo, lambdas, ou portas físicas. GMPLS vem sendo considerado a melhor proposta para integrar as tecnologias IP e WDM, primeiro porque ele pode ser usado como poderoso instrumento para engenharia de tráfego, e segundo porque ele é facilmente adequado à tecnologia WDM quando lambdas são usados como rótulos. No entanto, o seu mecanismo de sinalização em duas vias não é o adequado para o OBS.

Este artigo revê as principais questões relacionadas ao emprego da comutação baseada em rótulos e apresenta um conjunto de alterações nos mecanismos de sinalização e controle do GMPLS para melhor adequá-lo ao contexto da comutação de rajadas ópticas rotuladas. Em linhas gerais, propõe-se adequações na forma como os caminhos comutados por rótulos ("LSP – Label Switched Path") são estabelecidos. Sugere-se que a atribuição de rótulos no modelo sob demanda seja feita entre pares de roteadores consecutivos, o que difere da sinalização GMPLS tradicional, onde esta atribuição ocorre entre roteadores de entrada e saída da rede. Tal mudança tem o intuito de melhor aproveitar o mecanismo de sinalização sem confirmação utilizado pelo paradigma OBS.

Além desta seção introdutória, o artigo é composto de mais três seções. A seção II apresenta uma visão geral das redes totalmente ópticas, do MPLS e sua extensão GMPLS. A seção III descreve as modificações sugeridas na sinalização GMPLS para uma rede OBS e apresenta os resultados das simulações realizadas no *Network Simulator* (NS). A seção IV apresenta as conclusões gerais e sugere possíveis trabalhos futuros.

2. Redes Puramente Ópticas e a Comutação Baseada em Rótulos

Com o intuito de evitar os gargalos provocados pela necessidade de conversões dos sinais do domínio óptico para o domínio eletrônico, e vice-versa, nos dispositivos de comutação atuais (e.g. roteadores e comutadores), pesquisadores da área vêm investigando o uso de comutadores puramente ópticos para tornar possível explorar a capacidade total das fibras ópticas, que é em torno de 50 Tbps [ABE03b], através do uso de WDM.

No contexto das redes puramente ópticas, existem três abordagens possíveis [ABE02]: a comutação de lambdas, a comutação de pacotes ópticos (OPS - Optical Packet Switching) e a comutação de rajadas ópticas (OBS – Optical Burst Switching).

A comutação de lambdas se caracteriza por estabelecer um caminho óptico fim-a-fim dedicado enquanto durar uma determinada comunicação entre dois nós. A comunicação é realizada em três fases: estabelecimento de uma conexão, transmissão dos dados e liberação do circuito. Este tipo de comutação possui a vantagem de garantir o atraso fim-a-fim em uma rede, porém é ineficiente quando o tráfego não é intenso já que a capacidade do circuito é desperdiçada. Uma outra desvantagem diz respeito ao retardo provocado pelas etapas para o estabelecimento do circuito em redes bastante dinâmicas.

A comutação óptica de pacotes funciona enviando juntamente com os dados um cabeçalho que contém informações adicionais sobre o mesmo. Neste caso os pacotes são enviados nó a nó e o caminho que o pacote irá percorrer é definido em cada roteador. Uma rede com comutação óptica de pacotes pode resolver o problema da má utilização dos comprimentos de onda encontrados no caso anterior, uma vez que não há uma reserva exclusiva do canal. Além disso, não há o retardo introduzido pelo estabelecimento do canal. Porém, uma vez que a comutação de pacotes se baseia no mecanismo de encaminhamento via armazenamento e encaminhamento (store- and- forward), faz- se necessário o uso de buffers ópticos para armazenamento de luz em todos os dispositivos e as técnicas para armazenamento temporário de luz ainda não estão muito flexíveis [ABE03a].
Na comutação óptica de rajadas um pacote de controle (BCP – Burst Control Packet) é enviado primeiro em um canal separado e processado eletronicamente ao longo dos nós intermediários. Após certo tempo (suficiente para que os dispositivos processem o pacote de controle e se configurem), a rajada (tipicamente um conjunto de pacotes) é enviada sem a necessidade de confirmação do estabelecimento da conexão. Logo após o envio da rajada, o canal é liberado pra outras conexões. Caso haja algum problema na reserva dos recursos nos nós intermediários, a rajada será descartada. Este modelo vem merecendo atenção por se beneficiar das vantagens dos dois esquemas anteriores: utilização eficiente do canal, já que o estabelecimento de conexão é feito sem confirmação e o comprimento de onda é liberado tão logo a rajada seja enviada; e a não necessidade de buffers nos dispositivos ópticos, pois o paradigma se baseia no mecanismo de encaminhamento óptico, com a construção de um caminho de luz..
O uso do GMPLS vem recebendo destaque por estender o MPLS tradicional. O GMPLS se torna importante em redes ópticas uma vez que permite o uso de rótulos genéricos como comprimento de ondas, fatias de tempo e portas físicas. Uma outra característica importante é a possibilidade do nó iniciante da conexão sugerir os rótulos, o que agiliza o processo de sinalização. O uso do GMPLS permite a comutação rápida de pacotes e permite utilizar técnicas de engenharia de tráfego na rede.
A integração do GMPLS com o OBS permite o estabelecimento de caminhos não dedicados para as rajadas através de canais WDM ao longo dos nós. Além disso, para reduzir o tempo de processamento causado pelo roteamento no nível 3 (e.g. no nível IP), os BCPs podem ser comutados em nível 2 como ocorre no MPLS. Esta estrutura é chamada de OBS rotulada (LOBS – Labeled OBS) [QIA00].

3. Sinalização e Controle para Inter- Redes IP Baseadas em Redes Ópticas

Como destacado na Seção I, a interação do IP com WDM está oferecendo uma grande oportunidade para se reformular alguns aspectos da comunicação óptica. Contudo, para que se possa aproveitar melhor as características do paradigma OBS é necessário que os mecanismos de sinalização e controle do GMPLS sejam adaptados ao contexto da nova proposta de sinalização. Esta seção descreve o modelo de rede adotado neste trabalho, bem como apresenta as alterações sugeridas aos mecanismos de sinalização e controle do GMPLS.

3.1 Modelo Referência Adotado

O modelo de referência adotado neste trabalho consiste de roteadores IP/MPLS[1] conectados via inter- redes ópticas, através de caminhos de luz comutados dinamicamente (Figura 1). As redes ópticas que compõem estas inter- redes são baseadas nos paradigmas OBS e MPLS. A opção pela comutação de rajadas ópticas (OBS) deve- se tanto à sua maior eficiência, já que ela não necessita que lambdas fiquem dedicados a cada fluxo, como à sua maior adequação ao ambiente IP sobre WDM, uma vez que ela simplifica o processo de provisionamento dos recursos, aumentando a eficiência desses tipos de redes. Já a adoção do MPLS, como mencionado na Seção II, se justifica pela sua flexibilidade e capacidade de engenharia de tráfego, além da facilidade de adequação à tecnologia WDM, usando lambdas como rótulos.

[1] - Daqui em diante usar- se- á o termo MPLS de forma genérica para indicar o uso de MPLS, ou de seus aperfeiçoamentos MPλS ou GMPLS.

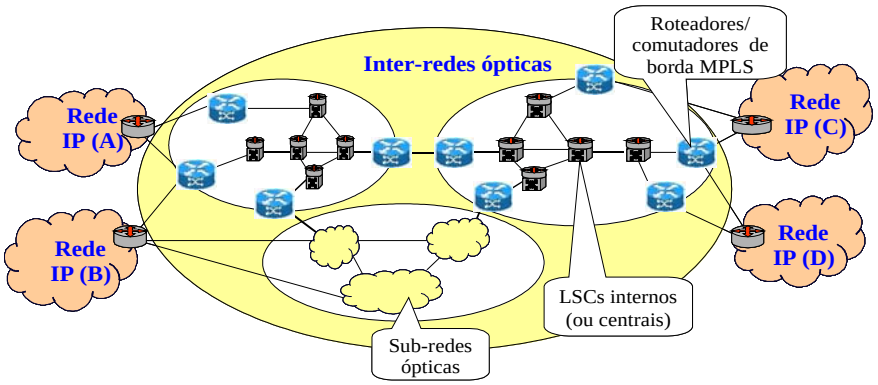


Figura 1. Modelo de rede adotado para a proposta, composto de múltiplos dispositivos de comutação ópticos (LSC) interconectados através de uma malha óptica.

Supõe-se que as inter-redes ópticas consistam de múltiplas redes ópticas, que possam ser administradas por diferentes entidades. Cada rede óptica pode ser formada por sub-redes compostas de múltiplos dispositivos de comutação óptica, aptos a realizar comutação por rótulos ("LSCs - Lambda Switching Crossconnects") e interconectados por enlaces ópticos em uma topologia geral. Esses LSCs podem ser equipamentos de diferentes fabricantes. Por questões de simplicidade, supõe-se também que existe um mapeamento um para um entre os controladores IP e os comutadores WDM.

A sinalização nas inter-redes ópticas é realizada fora de banda, existindo apenas um canal/lambda para sinalização de alta capacidade por fibra. As mensagens de sinalização são processadas eletronicamente por todos os nós, inclusive os internos. Os dados não sofrem qualquer tipo de processamento nos nós intermediários, assim como nenhuma suposição precisa ser feita sobre qual a taxa de transmissão dos dados. A inteligência da rede fica concentrada essencialmente nas bordas, e nenhum tipo de sincronização global é necessário.

Considera-se ainda que seja adotado o modelo de controle integrado, conforme apresentado em [RAJ04], onde tanto o domínio óptico como o domínio IP são gerenciados de forma unificada. Para estes dois domínios existe apenas uma instância de protocolo de roteamento e um único plano de controle baseado no MPLS. No caso de haver só um AS ("Autonomous System") envolvido, será considerado um único protocolo intra-domínio. Quando diversos ASs estiverem envolvidos, um protocolo de roteamento inter-domínio também deve ser usado. Ambos este protocolos precisam incluir as devidas extensões para tecnologias ópticas.

Nesta estrutura LOBS os nós da rede são classificados em dois grupos: nós internos (ou centrais) e nós de borda. Os nós internos comutam as rajadas com base em rótulos, e possuem funcionalidades semelhantes aos nós centrais LSR ("Label Switched Router") da arquitetura MPLS. Em LOBS cada BCP pode ser considerado um "rótulo jumbo", já que este deve conter, além dos rótulos, outras informações, tais como o tamanho da rajada e o tempo de ajuste entre o BCP e a rajada de dados.

Os nós de borda, por sua vez, possuem as funcionalidades eletrônicas pertinentes a roteadores IP e são os responsáveis pelo processo de montagem das rajadas, o qual pode envolver a definição de classes de equivalência ("FEC – Forwarding Equivalence Class") do MPLS, o empilhamento de rótulos e a agregação de LSPs. Durante a operação de montagem das rajadas, múltiplos pacotes IP são juntados em uma única rajada e o correspondente pacote de controle é construído. Além disso, diversos LSPs podem ser aglutinados em caminhos LOBS de maior capacidade, desde que seja respeitada a capacidade do canal.

3.2 Sinalização e Controle

Os protocolos associados ao MPLS precisam ser adaptados para trabalharem com OBS, mantendo a capacidade para engenharia de tráfego, e incorporando a eficiência da reserva sem confirmação de banda para as rajadas. Obviamente, os protocolos relacionados à sinalização usarão os canais de controle do OBS para trocar informações. Dentro da rede LOBS, a associação dos rótulos de entrada/saída de um LSP deve ser implementada usando a base de informação de rótulos ("LIB – Label Information Base") do MPLS. Além disso, como os BCPs contêm a informação necessária ao ajuste dos comutadores, entre as quais destacam-se o tempo de ajuste e o tamanho da rajada, eles devem ser enviados através de novas mensagens de controle do MPLS.

3.1.1 Atribuição de Rótulos

Para a comunicação ponto a ponto, a arquitetura MPLS define que a decisão de atribuir um rótulo específico para uma determinada FEC seja feita apenas pelo LSR de saída do enlace ("downstream") para o qual a associação está ocorrendo. Desta forma, o LSR de saída do enlace, após atribuir o rótulo para a FEC em questão ("downstream assigned"), informa o seu par na entrada do enlace ("upstream") sobre a associação. No que diz respeito à requisição das atribuições, a arquitetura MPLS permite duas variações, a sob demanda ("on- demand") e a não solicitada ("unsolicited"). Na variação sob demanda, o LSR de entrada do enlace requisita explicitamente ao seu par na saída do enlace um rótulo para uma determinada FEC, enquanto, na variação não solicitada, o LSR de saída do enlace realiza a atribuição sem solicitação prévia e informa seu par em seguida [ROS01].

Na nova proposta de sinalização sugere- se alterações no processo de atribuição de rótulos sob demanda. Quando um novo caminho comutado por rótulo (LSP) precisar ser estabelecido, a mensagem de atribuição de rótulos será enviada pelo roteador de borda responsável. Contudo, propõe- se que o nó de entrada de cada enlace faça uso da extensão proposta no GMPLS, sugerindo um rótulo, no caso um lambda, para o nó na saída do enlace e que o nó egresso de cada enlace confirme a aceitação do rótulo sugerido, assim que receber a mensagem.

Este comportamento difere do comportamento padrão do MPLS para controle ordenado onde a solicitação sob demanda de rótulos teria que percorrer todos os nós do caminho a ser estabelecido, até o LSR de saída da rede, que iniciaria a efetiva confirmação dos rótulos. A Figura 2 ilustra as etapas da solicitação/atribuição dos rótulos em cada enlace: o nó 1 envia o BCP para o nó2 ① e este último responde confirmando ou não a reserva dos recursos solicitados ②; enquanto a mensagem de resposta é enviada para o nó 1, o nó 2 envia o BCP para o próximo nó ③ e aguarda a resposta ④ e assim por diante ⑤ ⑥ até o nó de saída; caso não tenha ocorrido nenhum problema na atribuição do rótulo, a rajada começa a ser enviada pelo nó 1. Caso a reserva de recursos falhe em qualquer nó, a rajada será descartada

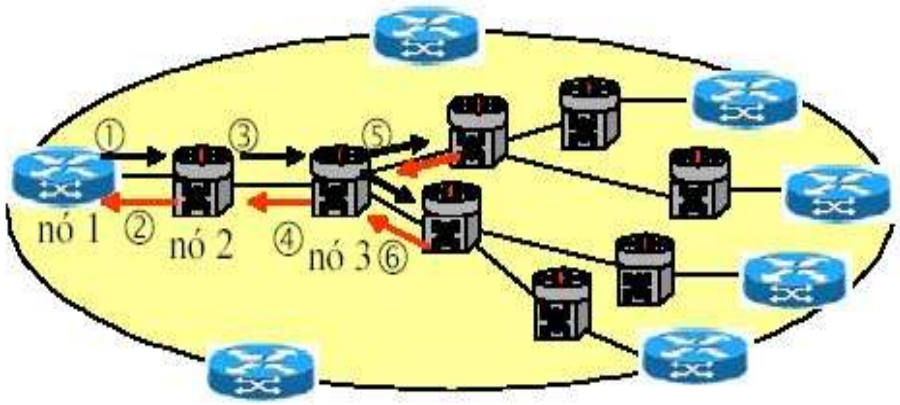


Fig. 2. Ilustração do funcionamento da proposta alternativa para atribuição de rótulos.

3.3 Engenharia de Tráfego e Outras Questões

Como os protocolos utilizados com o GMPLS já são essencialmente baseados em extensões de engenharia de tráfego ao MPLS, a princípio, é razoavelmente simples a adaptação ao contexto da engenharia de tráfego dos mecanismos relacionados à sinalização e controle sugeridos neste trabalho. O processo de sinalização pode ser basicamente o mesmo, com o processo de atribuição de rótulos sendo baseado nos esquemas sugeridos na Seção III.B. A maior diferença estaria no cálculo da árvore multiponto utilizando critérios (ou algoritmos) de engenharia de tráfego.

Adicionalmente, a flexibilidade da estrutura baseada em MPLS também possibilita algum tipo de ambigüidade na especificação dos caminhos LOBS, criando perspectivas interessantes nos esquemas de roteamento de caminhos LOBS. Por exemplo, parte das rotas pode ser especificada como "liberada", ou mesmo os nós de borda podem não especificar explicitamente todos os nós ao longo do caminho, mas apenas um subconjunto deles. Dessa forma, apenas parte do caminho LOBS seria previamente definido, deixando o restante para ser definido dinamicamente.

3.4 Avaliação dos Mecanismos de Sinalização e Controle Propostos

Para avaliar o impacto das modificações sugeridas por nós no processo de sinalização do MPLS no desempenho da rede, realizamos simulações no ambiente de simulação *NS* ("Network

Simulator") [NS04] comparando o modelo de sinalização proposto com o modelo tradicional de sinalização utilizado.

3.4.1 Extensões Desenvolvidas para o Simulador NS

O *NS* é uma simulador de redes orientado a eventos mantido pelo projeto VINT constituído por pesquisadores de instituições como UC Berkeley, USC/ISI, LBL e Xerox PARC.

O *NS* tem como principais vantagens o fato de ser gratuito e possuir código aberto permitindo a adição de novos módulos ou a alteração dos existentes de acordo com as necessidades do usuário. A versão do *NS* utilizada neste trabalho foi a 2.26. Para adicionar novas extensões ao simulador ou mesmo acrescentar novas características aos módulos já existentes é necessário ter um bom conhecimento da hierarquia de classes do *NS* e saber onde cada parte do código está situada dentro da árvore de diretórios do simulador [ABE01].

Para o nosso caso, foi necessário modificar as porções do código do *MNS* ("MPLS Network Simulator" - módulo do NS que trata de redes MPLS) responsáveis por tratar da requisição de um LSP, do mapeamento do LSP com um determinado rótulo e do estabelecimento do LSP usando a estratégia orientada à tráfego.

Após essas mudanças criou-se um novo esquema de estabelecimento de um caminho comutado por rótulo usando o controle ordenado, como descrito na seção 3.2. Esse esquema permite a sugestão de um rótulo ao nó egresso de cada enlace da rota que confirma ou não a aceitação do rótulo sugerido assim que recebe a mensagem de estabelecimento do LSP.

Porém, surgiu a necessidade de componentes que permitissem a simulação de redes ópticas baseadas na comutação OBS que pudessem ser utilizados em conjunto com o módulo MNS. Então se definiu que a etapa seguinte seria a construção de um novo agente (componente da arquitetura *NS* responsável por gerar e consumir pacotes além de implementar protocolos de várias camadas), que possuísse como principais funções: armazenar pacotes na borda da rede OBS, montar e desmontar rajadas, criar pacotes de controle e controlar os tempos de ajuste.

Além disso, modificou-se a estrutura do nó MPLS para interagir com o novo agente. Também foi realizada a substituição dos enlaces nativos do *NS* por enlaces que possuíssem características ópticas. Esses enlaces ópticos foram adaptados de uma extensão do *NS* chamada *OBS-ns* [DAW04].

3.4.2 Cenário utilizado para simulações

Como nossa proposta é essencialmente para as redes de backbone, optou-se por adotar uma topologia semelhante à de backbones existentes na Internet. Especificamente, a topologia escolhida foi inspirada no backbone da Abilene, como ilustrado na Figura 3.

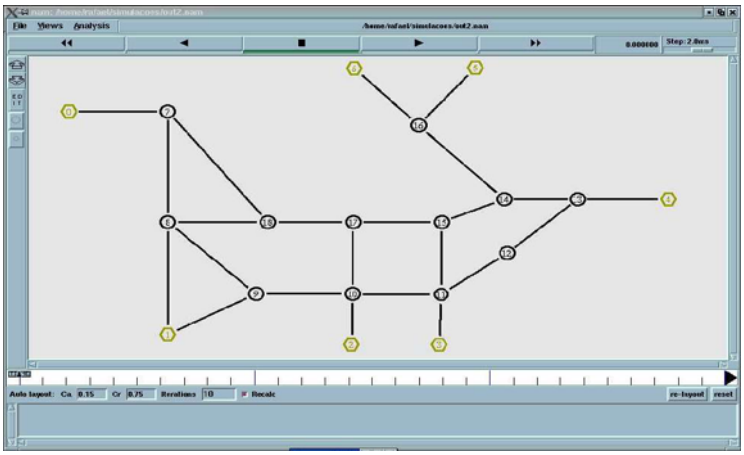


Fig.3. Topologia utilizada nas simulações

A rede possui 19 nós, dos quais 7 atuam como nós de borda e 12 atuam como nós centrais. Os nós de borda estão em formato de hexágono na cor amarela e são numerados de 0 a 6 e os nós internos estão em formato circular na cor preta e são numerados de 7 a 18. Utilizaram-se enlaces ópticos com capacidade de 10Gb possuindo retardos iguais (nas simulações esses retardos variam de 1 a 5ms para todos os enlaces). O tipo de tráfego usado foi o CBR e o tamanho das rajadas foi fixado em 2MB. Definiu-se que o tráfego teria como origem o nó 0 e o destino seria o nó 6. Também convencionou-se que após o envio de uma rajada o BCP da rajada seguinte é gerado e transmitido.

3.4.3 Resultados Obtidos

Considerando o modelo tradicional de sinalização do MPLS, o tempo entre o envio do BCP e o inicio do envio da rajada ocorre em T=P + ?, onde P é o tempo de propagação do BCP até o nó egresso da rede mais o tempo de resposta do estabelecimento do LSP e ? é tempo total de processamento do pacote de controle em todos os nós intermediários.

No modelo de sinalização proposto, o tempo entre o envio do BCP e a rajada é de T=? + δ, onde ? é o tempo de propagação do BCP até o primeiro nó do núcleo da rede mais o tempo de resposta do estabelecimento do LSP disparado por este nó e δ é o tempo de processamento do BCP neste nó (vide Fig. 2).

Levando em consideração esta análise partiu-se para a realização de simulações, utilizando o cenário descrito na seção anterior, para verificar o impacto das alterações promovidas na sinalização GMPLS no comportamento da rede. Um dos parâmetros usados para essa avaliação é o retardo total. O retardo total pode ser entendido como o tempo que leva desde o envio do BCP até quando a rajada chega ao nó de destino (em nosso caso, o nó egresso da rede). Outra medida de análise pode ser a quantidade de tráfego que passa na rede em um determinado instante de tempo.

A Figura 4 mostra a variação do retardo total e as Figura 5a a 5e mostram as quantidades de dados (em Megabytes) que trafegam na rede nos dois esquemas de sinalização abordados em ambientes de simulação onde todos os enlaces possuem o mesmo retardo (de 1ms a 5ms) em cada simulação.

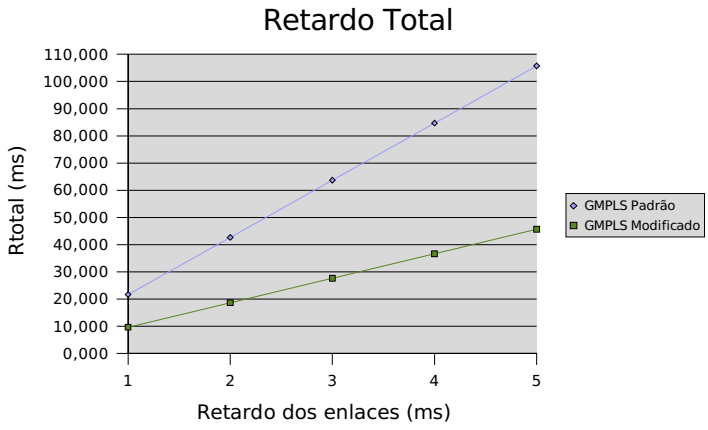


Fig. 4. Retardo Total

Pela Figura 4 percebe-se que na nova sinalização, o retardo total das rajadas é bem menor que na sinalização padrão. Isto ocorre, pois o tempo entre o envio do BCP e o envio da rajada é menor no modelo de sinalização proposto, uma vez que a rajada é enviada logo após a confirmação da atribuição do rótulo efetuada pelo primeiro nó do núcleo da rede.

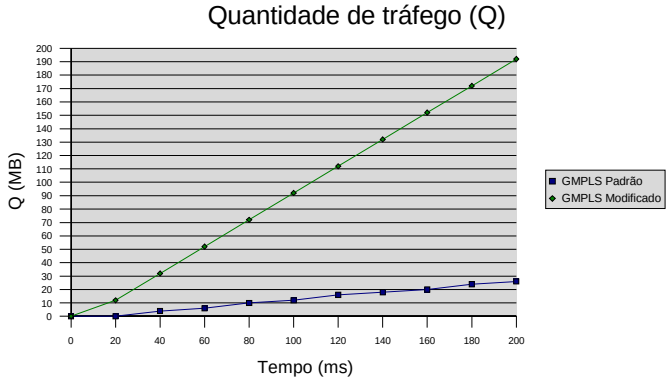


Fig.5a. Quantidade de tráfego (Utilização) com retardo dos enlaces de 1ms

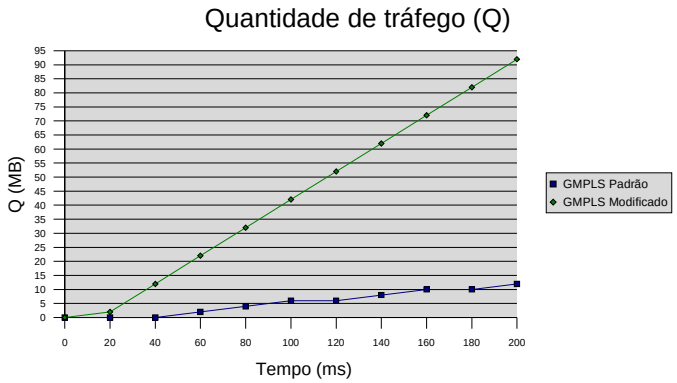


Fig.5b. Quantidade de tráfego (Utilização) com retardo dos enlaces de 2ms

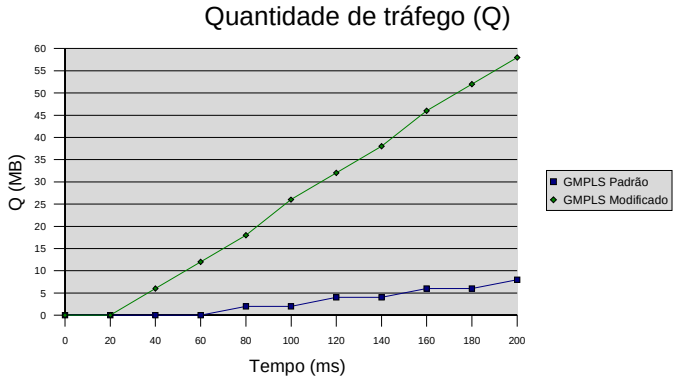


Fig.5c. Quantidade de tráfego (Utilização) com retardo dos enlaces de 3ms

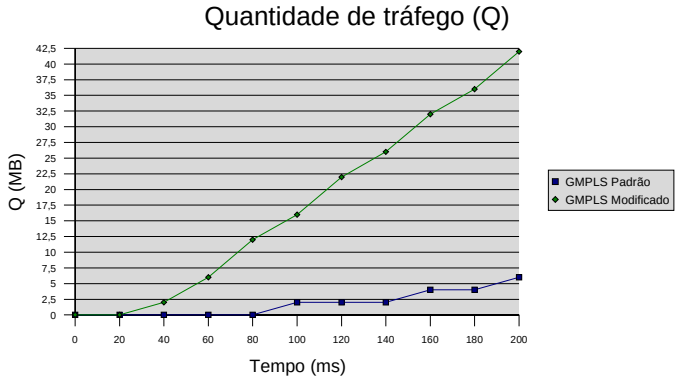


Fig.5d. Quantidade de tráfego (Utilização) com retardo dos enlaces de 4ms

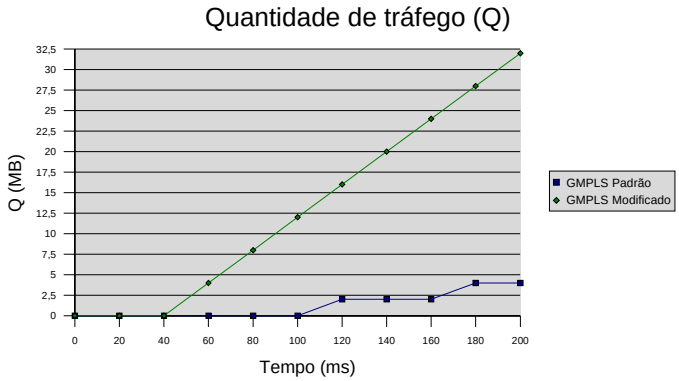


Fig.5e. Quantidade de tráfego (Utilização) com retardo dos enlaces de 5ms

As Figuras 5a a 5e mostram que no novo esquema a rede apresenta uma utilização maior, uma vez que ao longo do mesmo intervalo de tempo mais dados trafegaram no modelo de sinalização proposto, em relação ao modelo tradicional.
É possível também medir a taxa de utilização dos enlaces durante uma transmissão de rajadas ópticas. Essa medida pode ser interpretada como a relação percentual entre a quantidade de dados transmitida em um determinado intervalo de tempo e a capacidade máxima do neste mesmo intervalo. A Figura 6 mostra a variação dessa taxa de utilização em função do valor do retardo dos enlaces da rede óptica em um tempo total de simulação de 100 ms.
A partir da Figura 6 conclui- se que no modelo de sinalização proposto os recursos da rede são melhores aproveitados, pois a taxa de uso dos enlaces nesse modelo é maior que no esquema padrão. Isto se deve à maior quantidade de dados que podem ser transmitidos em uma rede com o esquema de sinalização proposto (vide Figuras 5a a 5e).

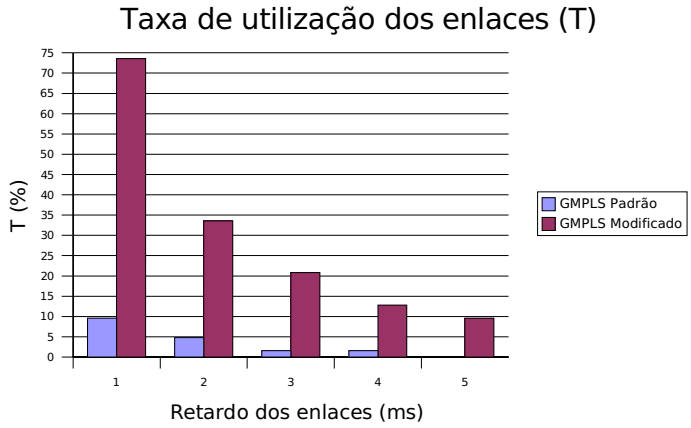


Fig. 6. Taxa de utilização dos enlaces em função do retardo dos enlaces

Levando em consideração que mais dados podem ser transmitidos no modelo de sinalização proposto, a Tabela I mostra as quantidades de rajadas (de 2MB) que podem ser enviadas em uma simulação de 100ms adotando em cada simulação os mesmos retardos em todos os enlaces.

TABELA I - NÚMERO DE RAJADAS ENVIADAS NOS DOIS ESQUEMAS DE SIMULAÇÃO

Retardo dos enlaces (ms)	Nº de rajadas enviadas (GMPLS Padrão)	Nº de rajadas enviadas (GMPLS Alterado)
1	6	46
2	3	21
3	1	13
4	1	8
5	0	6

Outras simulações foram realizadas com o objetivo de se analisar a utilização dos recursos da rede quando diferentes tamanhos de rajada são enviados.
A Figura 7 ilustra a porcentagem de uso dos enlaces estabelecendo- se tamanhos de rajada que variam de 2MB a 16MB onde o retardo dos enlaces é de 5 ms e o tempo total de simulação é de 100ms.

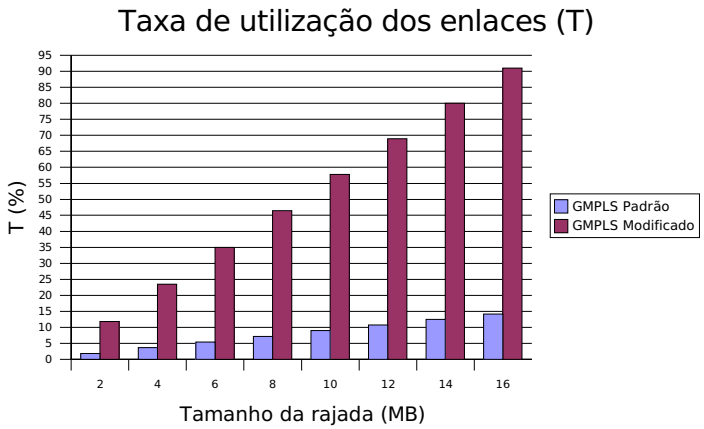


Fig. 7. Taxa de utilização dos enlaces (em função do tamanho das rajadas)

Pela Figura 7 observa-se que na sinalização sugerida há uma otimização do uso da capacidade dos enlaces da rede, conseqüência do menor retardo das rajadas e da maior vazão na rede obtida quando se adotam as alterações propostas nos mecanismos de sinalização e controle do GMPLS.

4. Conclusões e Trabalhos Futuros

Este trabalho discute os aspectos de sinalização e controle relacionados à interação do GMPLS com a comutação de rajadas ópticas (OBS) e apresenta sugestões de alterações no mecanismo de sinalização do GMPLS para melhorar o desempenho geral de redes ópticas. Especificamente, sugerimos adaptações na forma como os caminhos comutados por rótulos são disparados e nos mecanismos de distribuição e atribuição de rótulos. Em ambos os casos as modificações tiveram o intuito de melhor aproveitar as características do paradigma OBS, como por exemplo, o processo de reserva dos canais feito sem confirmação.

Verificou-se, através de simulações, que a proposta apresentada permite uma melhor utilização da rede uma vez que possui tempo de retardo de sinalização menor e conseqüentemente permite transferir mais dados em um determinado período de tempo.

Os trabalhos futuros incluem a implementação de mecanismos de engenharia de tráfego para a geração de caminhos alternativos para as rajadas. Um outro aspecto que deve ser analisado é a utilização do mecanismo de sugestão de rótulos a partir do nó *upstream* ou o envio da rajada sem ter recebido a confirmação do estabelecimento do LSP do primeiro enlace.

Referências

[ABE03b]- ABELÉM, A.; STANTON, M. "MIRROR: IP Multicast for Optical Burst-Switched Networks". *Revista da Sociedade Brasileira de Telecomunicações (SBT)*. Vol. 18, No. 2 , Outubro, 2003.

[ABE03a]- ABELÉM, A.; STANTON, M. "Análise da Proposta MIRROR: Conjunto de Adaptação ao IP Multicast para Redes Baseadas em Comutação Óptica." In: *Anais do 21 Simpósio Brasileiro de Redes de Computadores (SBRC2003)*. Natal, RN, Brasil. Maio, 2003.

[ABE02]- ABELÉM, A.; STANTON, M. A. "Inter-Redes IP Baseadas em Redes Ópticas." Livro texto dos Minicursos, SBRC2002, Cap. 2, pp. 63-123, Búzios, RJ, Brasil, Maio, 2002.

[ABE01]- ABELÉM, A. "Implementação de um Agente para Realizar Mapeamento entre Serviços Integrados e Serviços Diferenciados." Monografia, Departamento de Informática, PUC-Rio, Fevereiro, 2001.

[DAW04]- DAWN LAB. "Optical Burst Switching for Network Simulator". Url: http://dawn.cs.umbc.edu/owns/ . Acessado em: Janeiro, 2004.

[GUI98]- GUILEMOT, C. et al. "Transparent Optical Packet Switching: The European ACTS KEOPS project Approach". *IEEE Journal Lightwave Tech*. V. 16, No. 12, pp. 2117-2134, Dec., 1998.

[MAN03]- MANNIE E. (editor). "Generalized Multi-Protocol Label Switching (GMPLS) Architecture". *Internet Draft*, draft-ietf-ccamp-gmpls-architecture-07.txt.Maio, 2003.

[NS04]- NS. Network Simulator. Sitio do simulador. Url: http://www.isi.edu/nsnam . Acessado em: janeiro, 2004.

[QIA00]- QIAO, C. Labelled Optical Burst Switching for IP over WDM Integration. IEEECommunications Magazine. Vol. 38, no. 9, pp. 104- 114, Setembro, 2000.

[QIA99]- QIAO, C. e YOO, M. "Optical Burst Switching (OBS) – A New Paradigm for an Optical Internet". *Journal of High Speed Networks (JHSN).* Vol. 8, No. 1, pp. 69- 84, Agosto, 1999.

[RAJ04]- RAJAGOPALAN, B. et al., "IP over Optical Networks: A Framework". *RFC3717*. Março, 2004.

[ROS01]- ROSEN, E. et al. "Multiprotocol Label Switching Architecture". *RFC3031*. janeiro, 2001.

Gerenciamento da Integração de Processos de Software no APSEE-Integrate

Ana Vitoria Piaggio de Freitas, Anderson Baia Maia, Daltro José Nunes
Universidade Federal do Rio Grande do Sul, Instituto de Informática
Porto Alegre, Brasil, 91501-970
{avpfreitas, abmaia, daltro}@inf.ufrgs.br

Abstract
Software processes can be formally defined through process models, and executed by PSEEs - Process-Centered Software Engineering Environments. When the software development involves autonomous organizations, it is undesirable to use an unique process model to reflect the whole scope of the software process. PSEEs should provide infrastructure for processes that involve teams dispersed geographically. Such processes are usually quite extensive, consisting of several sub-processes, that can be defined in different modelling processes notations and executed in different PSEEs. PSEEs should be capable to interact, allowing interoperability among process models, in the modelling and execution levels. The goal of this article is to present the approach of the APSEE-Integrate environment for software processes integration management, in the modelling and execution levels. This approach brings, as main contributions, the flexibility during execution allied to the automated support to the integration of process models. The components of the environment are specified formally through Graph Grammars. This article discusses the components directly related to the management of processes integration models, that are: the processes interaction modeling language and the execution mechanism.

Keywords: Software Processes, Process-Centered Software Engineering Environments, Software Processes Integration.

Resumo
Processos de software podem ser definidos formalmente através de modelos de processo, e executados por ambientes de engenharia de software centrados no processo (PSEEs - *Process-Centered Software Engineering Environments*). Quando o desenvolvimento de software envolve organizações autônomas, é inviável utilizar um único modelo de processo para refletir todo o escopo do processo de software. PSEEs devem prover infra-estrutura para processos que envolvem equipes dispersas geograficamente. Tais processos são geralmente bastante extensos, consistindo de vários sub-processos, que podem ser definidos em diferentes notações de modelagem de processos e executados em diferentes PSEEs. PSEEs devem ser capazes de interagir, permitindo interoperabilidade entre modelos de processo, nos níveis de modelagem e de execução. Este artigo tem como objetivo apresentar a abordagem do ambiente APSEE-Integrate para gerência da integração de processos de software, nos níveis de modelagem e execução. Esta abordagem traz, como contribuições principais, a flexibilidade durante a execução aliada ao suporte automatizado à integração de modelos de processo. Os componentes do ambiente são especificados formalmente através de Gramáticas de Grafos. Este artigo discute os componentes diretamente relacionados à gerência de modelos de integração entre processos, que são: a linguagem de modelagem de interações entre processos e o mecanismo de execução.

Palavras-Chave: Processos de Software, Ambientes de Engenharia de Software Centrados no Processo, Integração de Processos de Software.

1. INTRODUÇÃO

O processo de desenvolvimento de software (ou simplesmente processo de software) pode ser compreendido como o conjunto de atividades a serem realizadas desde a concepção até a implantação do produto de software. Processos de software bem definidos reduzem o tempo de desenvolvimento e os custos de produção, além de possibilitarem melhor acompanhamento do desenvolvimento de software [5]. A experiência tem mostrado que processos de software têm grande influência na qualidade dos produtos de software; controlando-se o processo de software pode-se atingir melhor controle sobre a qualidade do produto final.

Processos de software podem ser definidos formalmente através de modelos de processo. O modelo de processo define a seqüência de atividades que devem ser realizadas, as ferramentas a serem utilizadas, os produtos e documentos que serão criados e/ou manipulados e os papéis das pessoas envolvidas no desenvolvimento de software [9]. Se notações formais forem utilizadas para a modelagem do processo, é possível automatizar partes do processo, suportar a interação e a cooperação entre os desenvolvedores e guiá-los durante a execução do processo. Os formalismos utilizados para descrever modelos de processo são chamados de linguagens de modelagem de processos.

A modelagem e a execução de processos de software é suportada por PSEEs (*Process-Centered Software Engineering Environment*s), que são ambientes de desenvolvimento de software orientados ao processo. A utilização de PSEEs traz benefícios, como melhor comunicação entre as pessoas envolvidas no desenvolvimento de software, realização de algumas atividades de forma automática, coleta de métricas, suporte à reutilização de modelos de processo e disponibilidade de informações sobre o andamento do processo de software [12]. Cada PSEE é caracterizado pela sua linguagem de modelagem, que deve suportar a descrição dos vários conceitos que caracterizam o processo de desenvolvimento de software, como [5]: atividades, artefatos de software, papéis, agentes humanos, recursos e ferramentas.

Devido à globalização da economia e ao rápido crescimento da Internet, cada vez mais processos de software se expandem por múltiplas organizações. Quando o desenvolvimento de software envolve organizações autônomas, é inviável utilizar um único modelo de processo centralizado para refletir todo o escopo do processo de software [13]. Segundo [9], PSEEs devem prover infra-estrutura para processos que envolvem equipes dispersas geograficamente, possuindo práticas de desenvolvimento distintas, usando diferentes ferramentas e linguagens de programação. Tais processos são geralmente bastante extensos, consistindo de vários sub-processos, que podem ser definidos em diferentes notações de modelagem de processos e executados em diferentes PSEEs. Embora os sub-processos sejam autônomos, eles precisam interagir. Atividades modeladas em um processo podem depender do estado de execução das atividades definidas em outros modelos de processo. PSEEs devem ser capazes de interagir, permitindo interoperabilidade entre modelos de processo, nos níveis de modelagem (integração entre modelos de processo especificados através de diferentes formalismos) e de execução (integração na execução realizada por diferentes PSEEs) [8].

Este artigo tem como objetivo apresentar a abordagem do ambiente APSEE-Integrate para gerência da integração entre processos de software, nos níveis de modelagem e execução. Esta abordagem traz, como contribuições principais, a flexibilidade durante a execução aliada ao suporte automatizado à integração de modelos de processo. O APSEE-Integrate é baseado em um meta-modelo, que especifica as informações dos modelos de processo envolvidos na integração, e em uma arquitetura que provê serviços para integração de PSEEs, com base no meta-modelo. É proposta uma linguagem visual de modelagem, um mecanismo para execução das relações de dependência entre processos e mecanismos para tradução de formalismos de modelagem dos PSEEs para o formalismo do APSEE-Integrate. A linguagem de modelagem utilizada no ambiente serve como base para o mecanismo de execução proposto. São utilizados métodos formais para especificação dos componentes do ambiente, sendo que serão destacados neste artigo apenas os componentes diretamente relacionados à gerência de modelos de integração entre processos, que são: a linguagem de modelagem de interações entre processos e o mecanismo de execução. O uso de Gramáticas de Grafos para especificação formal será discutido.

A seção 2 apresenta o APSEE-Integrate detalhando os componentes relacionados à modelagem e à execução de modelos de integração entre PSEEs. A seção 3 especifica formalmente a linguagem de modelagem e o mecanismo de execução. A seção 4 apresenta trabalhos relacionados encontrados na literatura. Conclusões são discutidas na seção 5.

2. APSEE-INTEGRATE

Esta seção apresenta a proposta para interoperabilidade entre PSEEs heterogêneos do APSEE-Integrate. Será apresentada uma visão geral da arquitetura do APSEE-Integrate, e em seguida o meta-modelo, a linguagem de modelagem e o mecanismo de execução serão detalhados informalmente.

2.1. Visão Geral

O objetivo principal do APSEE-Integrate é propor uma arquitetura para interoperabilidade de PSEEs heterogêneos. A arquitetura deve permitir comunicação e sincronização entre diferentes PSEEs, com comportamento consistente. O foco do trabalho é permitir que atividades pertencentes a diferentes modelos de processo, executando em PSEEs distintos, sejam interligadas através da definição de dependências entre as mesmas. Uma visão geral do APSEE-Integrate é apresentada na Figura 1, ilustrando as três camadas principais: meta-modelo, mecanismos para gerência da integração e tradução de formalismos.

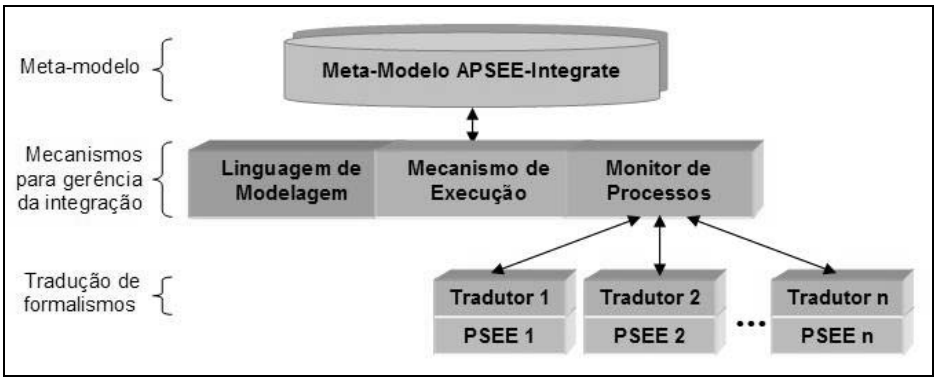


Figura 1 - Visão Geral do APSEE-Integrate

Uma breve descrição dos componentes do modelo é apresentada a seguir:

- **Meta-modelo**: define um formalismo comum, através do qual os modelos de processo poderão interagir.
- **Linguagem de modelagem**: linguagem para modelagem das dependências existentes entre atividades pertencentes a diferentes PSEEs.
- **Mecanismo de execução**: coordena a ativação das dependências entre atividades através da interpretação da linguagem de modelagem.
- **Monitor de processos**: monitora as alterações realizadas nos modelos de processo envolvidos na integração.
- **Tradutores**: são responsáveis pelo mapeamento dos formalismos de modelagem dos PSEEs para o meta-modelo do APSEE-Integrate. A cada formalismo de PSEE envolvido na integração está associado um tradutor específico, permitindo que o modelo de processo em execução no PSEE seja representado de acordo com o meta-modelo do APSEE-Integrate.

2.2. Meta-Modelo

Nesta seção serão apresentados os componentes do meta-modelo e seus relacionamentos através da notação UML. Esta notação é utilizada por ser largamente conhecida e facilitar o entendimento do modelo em alto nível de abstração.

O meta-modelo do APSEE-Integrate é baseado fortemente no meta-modelo de processos do APSEE [12]. Tal escolha foi realizada pelo fato de que a representação de dependências entre atividades do APSEE é bastante completa, permitindo expressar diversos tipos de dependências encontradas em processos reais. A Figura 2 mostra o diagrama de classes do meta-modelo APSEE-Integrate

A classe *Process* tem como objetivo representar os processos de software envolvidos na integração. Um processo da classe *Process* possui um identificador, um nome e um estado. O estado do processo pode ser: *Not_Started* (o processo ainda não foi iniciado), *Enacting* (o processo está em execução) ou *Finished* (o processo foi concluído).

Atividades são representadas através da classe *Activity*. Cada atividade possui identificador, nome, datas de início e término da execução, além de indicação de estado. O estado é o atributo que armazena a situação de cada atividade e serve de referência para a habilitação das dependências entre atividades. Os possíveis estados de uma atividade são descritos a seguir:

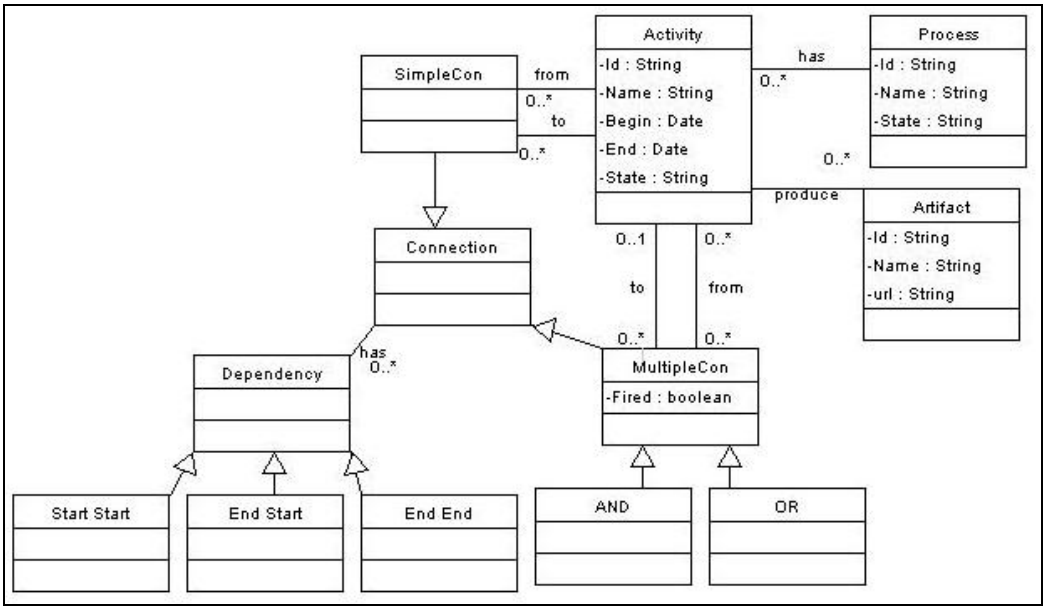


Figura 2 - Meta-modelo APSEE-Integrate apresentado através de um diagrama de classes UML

- *Waiting*: as pré-condições para execução da atividade definidas no modelo de processo no qual ela foi modelada ainda não estão satisfeitas;
- *Ready:* a atividade está pronta para começar;
- *Active:* a atividade está sendo realizada;
- *Finished:* a atividade foi concluída;
- *Canceled:* a atividade foi cancelada antes de iniciar;
- *Failed:* a atividade falhou após o inicio.

Atividades podem produzir artefatos de saída. A cada atividade podem estar associados artefatos, com a indicação da *url* (*unique resource location*) onde estão armazenados. O objetivo da definição de artefatos no APSEE-Integrate é permitir que os artefatos resultantes da execução de uma atividade estejam disponíveis para o PSEE onde ela foi modelada.

As conexões interligam atividades modeladas em diferentes processos, representando o fluxo de controle entre elas. São propostos fluxos de controle simples (com uma atividade origem e uma atividade destino) e múltiplos (envolvendo várias atividades origem e uma atividade destino).

As conexões possuem um tipo de dependência (*end-start*, *start-start*, *end-end*) que influencia diretamente na execução de atividades e pode representar diferentes dependências encontradas em processos reais. Conexões múltiplas podem ser combinadas com operadores lógicos (AND, OR). Os tipos de conexão e de dependência são detalhados a seguir:

- Conexão simples *(SimpleCon)*: define o fluxo de controle entre duas atividades (origem e destino). Este tipo de conexão indica que a atividade destino depende da atividade origem. Os tipos de dependência entre atividades podem ser:
  a) *End-Start*: a atividade destino será iniciada quando a origem terminar;
  b) *Start-Start*: a atividade destino será iniciada quando a origem for iniciada;
  c) *End-End*: a atividade destino será finalizada quando a origem terminar.

- Conexão múltipla (*MultipleCon*): possui várias atividades ou conexões múltiplas como origem e uma atividade ou conexão múltipla como destino. A semântica da conexão depende do operador lógico associado (AND – todas, OR – pelo menos uma) e do tipo de dependência utilizado (*end-start*, *start-start*, *end-end*). Uma aplicação desta conexão seria que assim que todas as atividades terminarem, então uma atividade destino pode começar (operador lógico AND e tipo de dependência *end-start*). Conexões múltiplas possuem o atributo *Fired*, que indica se a conexão já foi ou não disparada.

2.3. Linguagem de Modelagem

A modelagem da integração entre PSEEs requer formalismos de alto nível para descrever aspectos de coordenação entre atividades. Estes formalismos devem ser convenientes não somente para a representação, mas também para a execução. Com base nisso é proposto um formalismo de modelagem para o APSEE-Integrate que permite representação gráfica das dependências entre atividades com base no meta-modelo mostrado anteriormente.

Nesta seção será apresentada uma descrição informal da notação da linguagem e exemplos de seu uso. Os símbolos visuais da linguagem de modelagem são ilustrados na Figura 3.

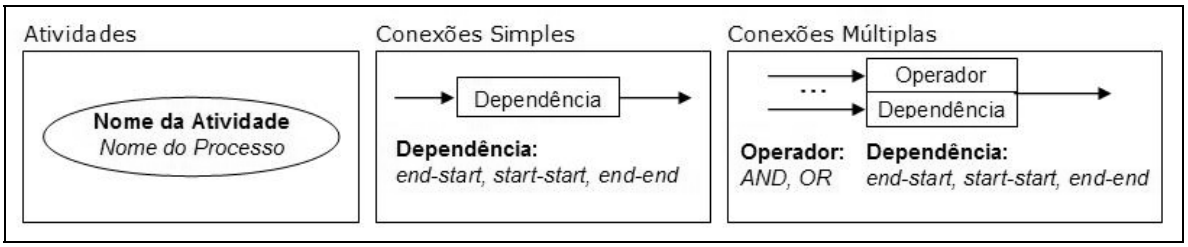


Figura 3 – Símbolos visuais da linguagem de modelagem

A definição de atividades é feita de forma automática, a partir das atividades existentes nos modelos de processo que participam da integração. Não podem ser inseridas novas atividades no APSEE-Integrate; pode-se apenas relacionar, através de conexões, as atividades existentes nos modelos de processo.

As conexões relacionam atividades modeladas em diferentes processos. O APSEE-Integrate, com base nas conexões definidas, inicia ou finaliza a execução de atividades. A conexão simples tem inicio e fim em atividades diferentes, e o nodo que a representa é um retângulo contendo o tipo de dependência escolhido (*end-start, start-start ou end-end*). Uma restrição importante é que a definição de uma conexão simples não pode inserir ciclos nas dependências entre processos, ou seja, não pode haver um fluxo de controle no sentido contrário ao sentido da conexão simples entre as duas atividades envolvidas.

A representação de conexões múltiplas é feita através de um retângulo dividido, onde a parte superior indica o operador lógico associado (AND, OR), enquanto a parte inferior indica o tipo de dependência (*end-start, start-start ou end-end*). Por definição, uma conexão múltipla pode ter vários antecessores e somente um sucessor. Os antecessores e sucessores de conexões múltiplas podem ser atividades ou conexões múltiplas. Portanto, o fluxo de controle entre processos pode ser definido mais detalhadamente através da combinação de várias conexões múltiplas encadeadas.

Um exemplo de definição de conexões entre atividades é ilustrado na Figura 4. Neste exemplo, a *Atividade C*, modelada no *Processo 1*, será iniciada após o término (dependência *end-start*) da *Atividade A*, modelada no *Processo 2*, e da *Atividade B*, definida no *Processo 3* (operador lógico AND), ou após o término da *Atividade E*, definida no *Processo 3* (operador lógico OR). Quando a *Atividade D*, definida no *Processo 2*, for finalizada, a *Atividade C* também será concluída (dependência *end-end*).

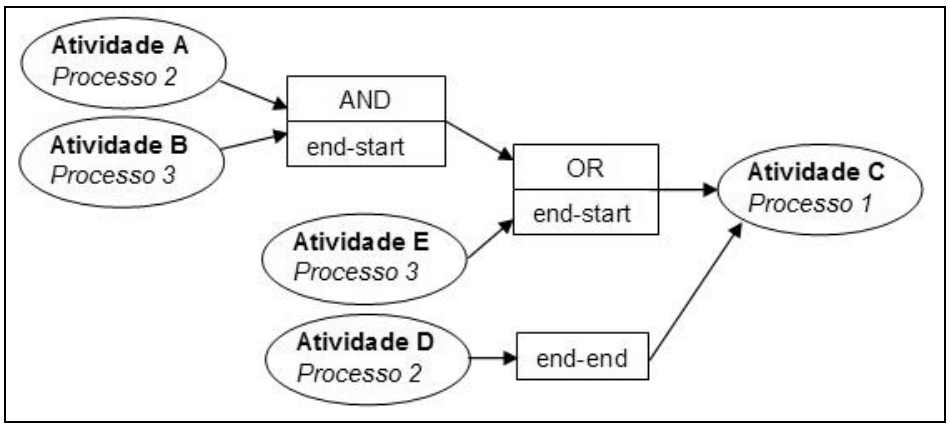


Figura 4 - Exemplo de definição de conexões entre atividades

2.4. Mecanismo de Execução

Durante a execução do modelo de integração, atividades são iniciadas e terminadas, de acordo com as conexões existentes e com o estado das atividades em execução nos diferentes PSEEs. O Mecanismo de Execução interpreta o modelo de integração e gerencia o estado de execução das atividades, com base nos modelos de processo distribuídos. A execução do modelo de integração tem como principais características:

- Permitir a execução de modelos de integração incompletos e modificação dinâmica: o modelo de integração pode continuar sendo modelado enquanto algumas conexões já foram ativadas. A modelagem também pode ocorrer em partes do modelo que já iniciaram. Nesse caso, mudanças dinâmicas são permitidas, desde que não afetem a consistência do modelo de integração. A criação de um meta-modelo unificado e a construção de um mecanismo de execução que interage com as fases de modelagem e execução possibilita essa característica;

- Monitorar as alterações realizadas nos modelos de processo distribuídos, para garantir a integridade das conexões existentes entre os modelos. A exclusão de uma atividade pode implicar na alteração de conexões;

- O mecanismo de execução fornece a semântica da linguagem de modelagem. Essa semântica é definida formalmente através de Gramática de Grafos. A especificação formal da semântica da linguagem de modelagem permite tratar o problema em alto nível de abstração, provê uma base para rastreamento e análise de impacto de alternativas, e define as conseqüências das transições de estado de forma resumida e simplificada. Essa semântica será apresentada na seção 3;

- Permitir visualização da execução através do formalismo gráfico e executável para modelagem de conexões entre processos. O mesmo formalismo é usado para modelar as conexões e acompanhar sua evolução.

A execução verifica continuamente o estado de execução dos modelos de processo, de modo a tratar as alterações no modelo e a evolução do estado de execução. A seguir é ilustrado o comportamento da execução em virtude dessas modificações para atingir os objetivos citados do APSEE-Integrate. Deve-se observar que, para uma compreensão geral da execução, é necessário levar em consideração a semântica das conexões entre atividades.

A Figura 5 apresenta um diagrama de transição para os possíveis estados de uma atividade, sendo que os estados e seus significados foram apresentados na seção 2.2. Os estados *Waiting* e *Canceled* não foram incluídos no diagrama porque eles são determinados apenas pelo modelo de processo no qual a atividade foi modelada. O APSEE-Integrate não define transições com origem ou destino nestes estados. Eles refletem a evolução da execução dos modelos de processo, e são utilizados pelo Mecanismo de Execução para garantir consistência nas conexões entre atividades.

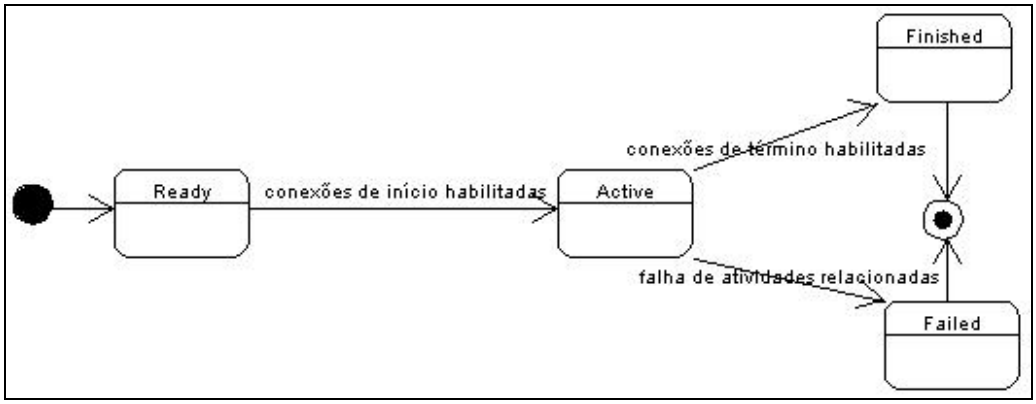


Figura 5 - Transição de estados de uma atividade

O mecanismo de execução verifica as mudanças de estado ocorridas nos modelos de processo, e dispara transições quando necessário. Quando todas as conexões relacionadas ao inicio de uma atividade (dependências *end-start* e *start-start*) forem habilitadas, então o estado de execução da atividade será modificado para *Active*. Já a habilitação das conexões de término (dependência *start-end*) implica na alteração do estado de execução da atividade para *Finished*.

As atividades de um modelo de processo podem falhar durante sua execução. Quando uma atividade falha e existem conexões nas quais ela é origem, as atividades destino irão falhar somente se dependerem do fim da atividade que falhou. Daí, conexões *start-start* não propagam falhas. A falha é propagada para todas as sucessoras que dependem da atividade, inclusive as que ainda não iniciaram. Conexões múltiplas OR só propagam falhas caso todas as atividades origem tenham falhado.

3. SEMÂNTICA FORMAL DO APSEE-INTEGRATE

A seção anterior descreveu informalmente o meta-modelo, a linguagem de modelagem e o mecanismo de execução propostos para o APSEE-Integrate. Esta seção utiliza Gramática de Grafos para apresentar a sintaxe da linguagem de modelagem e a semântica do mecanismo de execução. A definição formal do meta-modelo não será apresentada neste artigo. Os componentes do meta-modelo já foram apresentados na seção 2.2 através de diagrama de classes UML.

A partir das características do problema, foi realizada uma avaliação de métodos mais adequados para especificação. Considerando que o trabalho está inserido no grupo de pesquisa PROSOFT, coordenado pelo Prof. Dr. Daltro José Nunes, que propõe um ambiente de desenvolvimento formal de software de mesmo nome, foi avaliada a definição das propostas para funcionamento integrado ao ambiente.

A evolução do ambiente PROSOFT está intimamente ligada com o objetivo de construir ferramentas que apóiem o uso de métodos formais no ciclo de vida do software. Além disso, o ambiente é baseado em objetos e distribuído, o que facilita a integração de ferramentas que necessitam desse recurso. No PROSOFT, as especificações formais constituem uma base sólida que guia a implementação de protótipos ou sistemas de software completos. Nesse ambiente as ferramentas são construídas utilizando um paradigma algébrico, e há uma correspondência entre os componentes especificados algebricamente e a sua implementação. Assim, o formalismo algébrico mostra-se adequado para especificar os tipos de dados e operações básicas do meta-modelo aqui proposto.

Para a especificação da linguagem de modelagem visual e do mecanismo de execução foi escolhida a abordagem de Gramática de Grafos. A escolha foi inspirada no trabalho proposto por Lima Reis [12], onde se utiliza esse formalismo para especificar a sintaxe e a semântica da linguagem visual e do mecanismo de execução do APSEE. O fato de Gramáticas de Grafos serem formais e intuitivas ao mesmo tempo, e de poderem tratar com simplicidade aspectos de concorrência e distribuição de sistemas influenciou na sua escolha para complementar a especificação da semântica da execução de processos.

Gramáticas de grafos baseiam-se no processo de transformação que um grafo pode sofrer em função de um conjunto de regras previamente definidas. Um sistema é especificado em termos de estados, que são modelados por grafos, e de mudanças de estados, modeladas por regras ou derivações. A aplicação de uma regra a um grafo G é chamada passo de derivação, e isso só é possível se existe uma ocorrência do lado esquerdo (L) da regra no grafo atual G. O lado direito (R) da regra define o grafo resultante da aplicação desta regra. A interpretação de uma regra r:L → R é feita da seguinte forma:

- Itens em L que não têm imagem em R são eliminados;
- Itens em L que são mapeados para R são preservados;
- Itens em R que não possuem uma pré-imagem em L são criados.

Existem várias abordagens diferentes para Gramáticas de Grafos. Usaremos neste trabalho a abordagem algébrica através de mecanismos de tipagem nos grafos. Será apresentado um grafo tipo representando os tipos de nodos e arcos do sistema. O grafo inicial e os grafos derivados das transformações devem ser compatíveis com o grafo-tipo. Em seguida serão apresentadas algumas regras de derivação do grafo inicial.

3.1. Sintaxe da Linguagem de Modelagem

Para desenvolver uma linguagem visual é necessária inicialmente uma especificação desta linguagem. Devido ao caráter multidimensional de linguagens visuais, suas especificações textuais são normalmente difíceis de escrever e entender. Por isso, especificações visuais de linguagens visuais têm sido cada vez mais utilizadas. Uma técnica recente para especificar linguagens visuais foi proposta por Bardohl em [1] e [2]. Nesta abordagem, chamada GENGED, uma linguagem visual é especificada por um alfabeto e uma gramática. De acordo com Bardohl, o uso de gramáticas de grafos provê um formalismo mais natural e visual para a especificação de linguagens visuais.

Segundo Bardohl em [2], existem dois níveis de descrição de linguagens visuais: o nível de sintaxe abstrata, que descreve o significado lógico das sentenças e o nível de sintaxe concreta, que é usado para descrever o layout das sentenças. A combinação dos dois níveis é chamada Sintaxe Visual. A sintaxe abstrata geralmente corresponde a um grafo tipo onde os vértices são os elementos do alfabeto da linguagem e os arcos denotam o relacionamento entre eles. Para complementar a sintaxe visual, devem ser definidas regras para especificar como são construídas as sentenças visuais e, quando necessário, sua semântica.

A especificação da linguagem de modelagem do APSEE-Integrate foi inspirada na abordagem GENGED [2]. O grafo-tipo da linguagem de modelagem forma um diagrama que conecta vários componentes da linguagem. A Figura 6 mostra o diagrama construído para definir a sintaxe da linguagem Os símbolos do visuais do grafo-tipo são os mesmos da linguagem de modelagem, apresentados na seção 2.3.

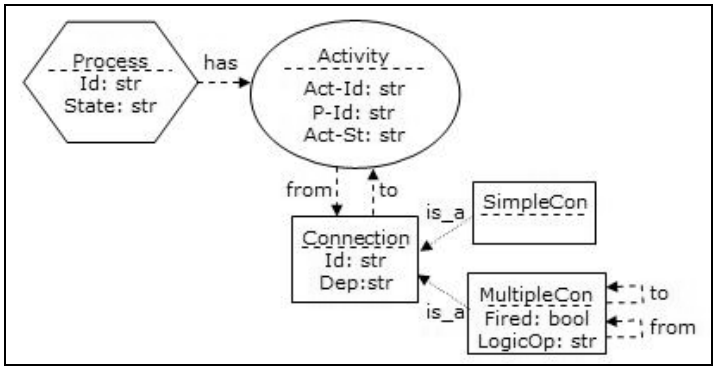


Figura 6- Grafo-tipo da linguagem de modelagem

Cada símbolo está associado a um nome e um conjunto de atributos (separados por uma linha do nome do símbolo). Atributos são elementos tipados usados para representar o estado do processo. As setas do diagrama representam relacionamentos, e correspondem aos arcos do grafo.

No grafo-tipo, um processo de software é composto por atividades. Os arcos *from* e *to* descrevem a origem e o destino das conexões. O relacionamento *is_a* pode ser compreendido como herança, similar ao uso de herança em linguagens de programação orientadas a objetos. O relacionamento *is-a* simplifica bastante o grafo-tipo e as regras de transformação correspondentes, já que atributos e relacionamentos são herdados aos subtipos. Conexões possuem o atributo *dep*, que representa o tipo de dependência (*end-start*, *start-start*, *end-end*). Conexões múltiplas possuem ainda indicação do operador lógico (AND ou OR), através do atributo *LogicOp*.

Após a definição do grafo-tipo, é necessário declarar como criar sentenças válidas para a linguagem proposta. Na abordagem GENGED [2] é proposta a criação de uma gramática visual para gerar as sentenças da linguagem e um conjunto de regras de comportamento para definir as transformações que modificam o sistema e os estados do modelo. A gramática visual da linguagem do APSEE-Integrate é responsável por gerar instâncias de modelos de integração entre processos e também garantir sua consistência.

Foram descritas regras para a gramática que gera sentenças da linguagem. Em muitos casos, foram usadas condições negativas de aplicação (NAC – *Negative Application Conditions*) para descrever condições que devem ser negativas para que a regra seja habilitada. Nesse caso as partes cortadas no lado esquerdo da regra formam as condições negativas e não devem estar presentes na instância do grafo para que a regra seja aplicada.

A Figura 7 mostra um exemplo de regra para inserir uma conexão simples entre duas atividades. Na figura, o lado esquerdo da regra contém uma situação onde existem duas atividades, *act_id1* e *act_id2*, pertencentes a processos distintos (*p_id1* e *p_id2*), e acima da figura é mostrada uma chamada que corresponde à solicitação do usuário. A regra da figura é disparada se o usuário solicita a inclusão de uma conexão simples, com uma determinada dependência, entre duas atividades existentes, *act_id1* e *act_id2*. Para prevenir a inclusão de ciclos - que causariam *deadlock* – uma NAC define que a atividade destino não pode ter um fluxo de controle direto ou indireto para a atividade origem, enquanto uma NAC adicional assume que não existe nenhuma conexão de controle entre a origem e o destino. Como resultado, se a regra for aplicada, é acrescentada uma conexão simples entre as atividades. Vale notar que esta regra se aplica caso a atividade origem não tenha sido cancelada ou falhada (não se devem definir conexões entre atividades nesses estados). A atividade destino deve estar no estado *waiting* ou *ready*, ou seja, não pode ter sido iniciada. Esta condição é necessária porque caso a

atividade origem ainda não tenha sido iniciada, se a atividade destino estiver executando (estado *active*), a dependência da conexão terá sido violada.

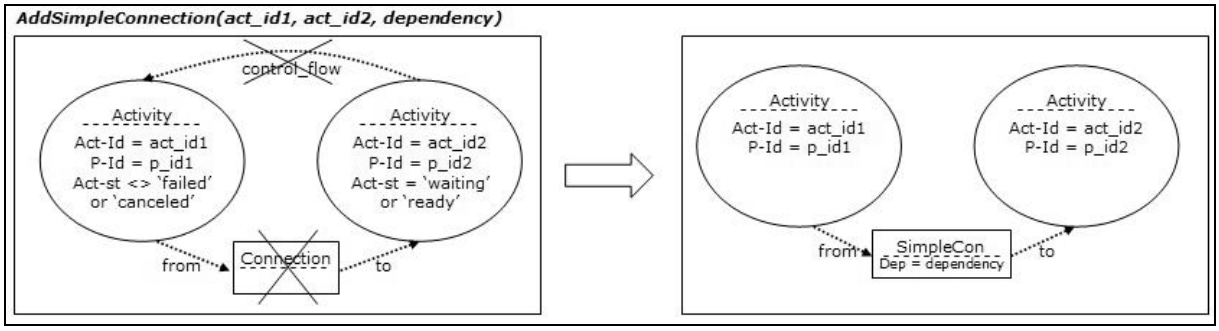


Figura 7 - Regra para inclusão de uma conexão simples entre atividades

A Figura 8 ilustra uma regra para definir o destino de uma conexão múltipla. A regra indica que, dadas uma atividade e uma conexão múltipla, a atividade deve ser definida como destino da conexão. Para isto, é necessário que a atividade não possua conexão de destino pré-existente com a conexão; não exista fluxo de controle da atividade para a conexão (para evitar ciclos) e nenhuma atividade ou conexão tenha sido definida anteriormente como destino da conexão. Além disso, o estado da atividade é determinado do lado esquerdo da regra.

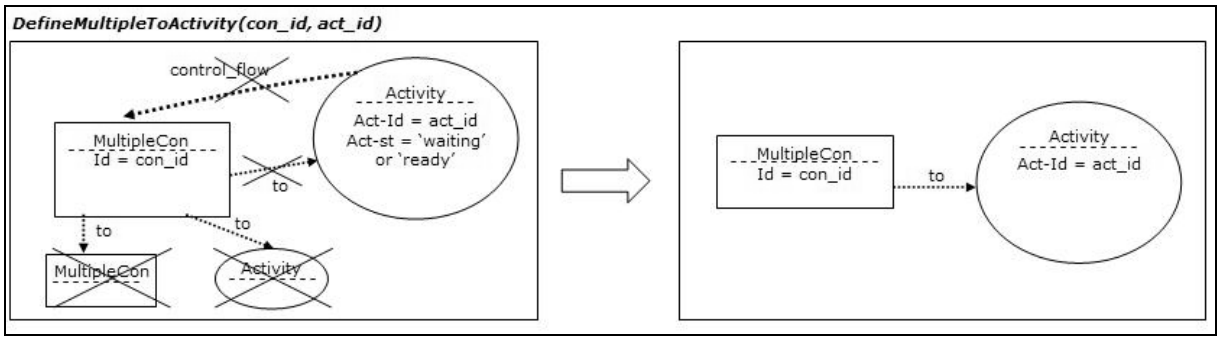


Figura 8 - Regra para definir uma atividade como destino de uma conexão múltipla

Na Figura 9 é apresentada uma regra para definir uma conexão múltipla (*FromCon_id*) como destino de outra (*con_id*). Os pré-requisitos para aplicação da regra são a não-existência de relação de destino entre as conexões e a não-existência de fluxo de controle da conexão origem para a conexão destino. A regra também indica que a conexão destino não pode ter sido disparada (*fired = false*).

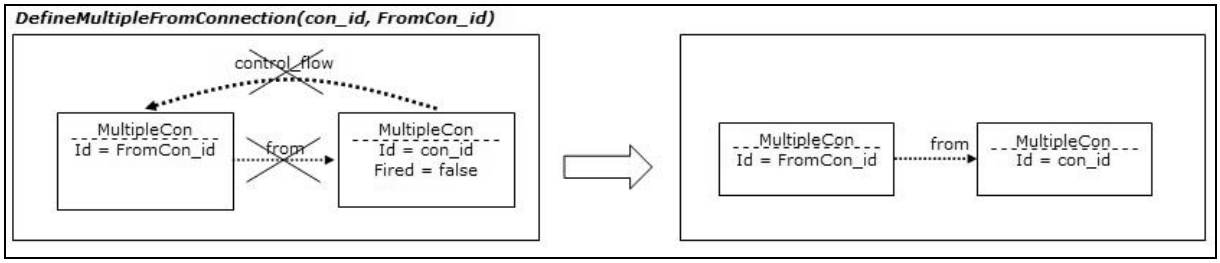


Figura 9 - Regra para definir uma conexão múltipla como origem de outra conexão múltipla

Além de regras relativas à definição de conexões, foram estabelecidas regras para testar a existência de fluxos de controle entre atividades e conexões múltiplas. O conjunto de regras definidas garante a não-inclusão de deadlocks, mantendo o modelo de integração em um estado consistente.

3.2. Semântica de Execução

Nesta seção serão apresentadas regras desenvolvidas para dar semântica à execução dos modelos de integração entre processos definidos segundo a linguagem de modelagem do APSEE-Integrate. Foram criadas regras para definir o comportamento do modelo de integração durante a execução.

A Figura 10 apresenta uma regra que define como é feita a transição de estado de uma atividade de *ready* para *active*. Para que a transição ocorra, a atividade não pode depender com conexão simples *start-start* de uma atividade que ainda não começou; não pode depender com conexão simples *end-start* de uma atividade que ainda não terminou; não pode ser destino de nenhuma conexão múltipla ainda não disparada. Além disso, é necessário que a atividade seja destino de uma conexão simples *start-start* que tenha como origem uma atividade que já começou.

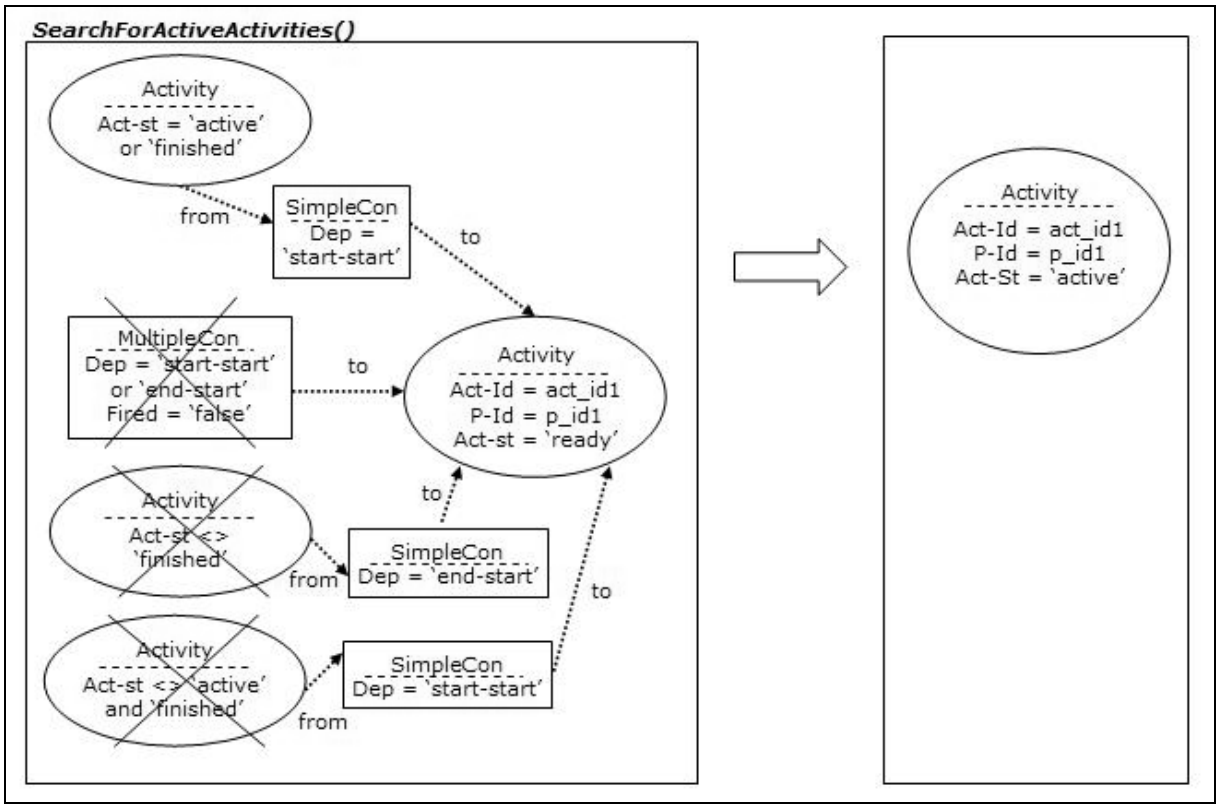


Figura 10 - Regra para transição de estado de uma atividade

Além das regras que procuram atividades prontas para executar, foram definidas regras para buscar atividades que podem ser finalizadas; para propagar falhas entre atividades; para tratar ativação de conexões múltiplas (definem quando o atributo *fired* será igual a *true*). O conjunto de regras obtido define a semântica da execução de modelos de integração entre PSEEs no APSEE-Integrate.

4. TRABALHOS RELACIONADOS

Dentro da comunidade de Processos de Software, a área de pesquisa em integração de processos de software descentralizados tem-se tornado um tópico bastante abordado [13]. Soluções têm sido propostas para tratar a interoperabilidade nos níveis de modelagem e execução de modelos de processo. Esta seção apresenta alguns dos principais trabalhos encontrados na literatura, comparando-os com a abordagem do APSEE-Integrate.

Process Landscaping [10] é um método que descreve o relacionamento entre modelos de processo distribuídos. O foco maior está na fase de definição de requisitos, e o objetivo é reduzir a complexidade dos modelos de processo. Esta abordagem preocupa-se com o nível de modelagem, não abrangendo a execução, e faz a suposição implícita de que existe um único sistema por trás de todos os modelos de processo.

Provence e Endeavors são exemplos de soluções que apresentam heterogeneidade no nível de execução, sem prover interoperabilidade no nível de modelagem [3]. Provence é um ambiente heterogêneo com o objetivo de prover suporte não-intrusivo para a execução de projetos. A interoperabilidade obtida é *ad hoc* e em baixo nível de abstração, sem definição de modelo de interoperabilidade. Endeavors é um sistema de *workflow*/processos distribuído, que suporta execução heterogênea de *handlers* de baixo nível, possivelmente escritos em diferentes linguagens de programação.

A interoperabilidade entre PSEEs homogêneos é suportada por ferramentas como Oz [4], PROSYT [6] e Serendipity-II [11]. A essência do modelo de interoperabilidade do Oz reside em dois mecanismos para

abstração da definição e da execução interprocessos: *treaties* – que permitem a definição de sub-processos compartilhados; e *summits* – que provêem a execução dos *treaties* definidos. O PROSYT integra comunicação baseada em eventos com suporte a mobilidade de código. E o Serendipity-II suporta modelagem distribuída e cooperativa de processos de software. Modelos de processo são replicados entre os usuários, e diferentes versões podem ser executadas.

Ambientes como APEL [8], PIE [7] e CAGIS [14] suportam interoperabilidade de PSEEs heterogêneos. APEL e PIE têm foco na integração de ferramentas distintas, não necessariamente PSEEs. Ambos baseiam-se na idéia de um modelo de processo comum, que define a integração entre as diferentes ferramentas. O modelo de processo comum é executado, e coordena a execução das demais ferramentas. O CAGIS interliga ferramentas de *workflow*, utilizando agentes de software para realizar atividades que envolvam mais de um *workflow*.

O modelo apresentado neste artigo suporta modelagem de processos de software distribuídos e execução descentralizada envolvendo diferentes PSEEs. Assim como no APEL e no PIE, o modelo proposto controla a integração entre PSEEs. Mas tal controle é realizado de forma transparente, e cada PSEE executa de forma autônoma. Enquanto o foco do CAGIS é suportar atividades que envolvam mais de uma equipe para sua execução, o modelo apresentado neste artigo visa suportar a integração entre atividades inter-relacionadas executadas de forma independente pelas equipes.

5. CONCLUSÕES

PSEEs devem prover infra-estrutura para processos que envolvem equipes dispersas geograficamente, possuindo práticas de desenvolvimento distintas, usando diferentes ferramentas e linguagens de programação. PSEEs devem ser capazes de interagir, permitindo interoperabilidade entre modelos de processo, nos níveis de modelagem (integração entre modelos de processo especificados através de diferentes formalismos) e de execução (integração na execução realizada por diferentes PSEEs) [8].

O artigo propõe uma abordagem para integração de processos, modelados e executados em diferentes formalismos. O modelo proposto, o APSEE-Integrate, permite a definição de dependências entre atividades, de modo que a execução de modelos de processo seja integrada.

Foram apresentados o meta-modelo de integração de processos, a linguagem de modelagem e o mecanismo de execução. Além da descrição informal, a sintaxe formal da linguagem e a semântica formal do mecanismo de execução foram apresentadas, utilizando gramática de grafos.

Neste artigo, inicialmente foi apresentada uma visão geral do APSEE-Integrate. Em seguida foram apresentados informalmente o meta-modelo, a linguagem de modelagem e o mecanismo de execução. A semântica formal da linguagem de modelagem e do mecanismo de execução foi discutida. A abordagem para integração de PSEEs apresentada traz como principais contribuições:

- A semântica formal do mecanismo de execução através do uso do paradigma PROSOFT-Algébrico combinado com Gramática de Grafos. A especificação formal permite trabalhar em alto nível de abstração, provendo uma base para rastreamento, verificação formal e permite definir as conseqüências das transições de estado de forma resumida e unificada;

- A importância dada aos requisitos resultantes do envolvimento humano no processo, aumentando a flexibilidade do sistema resultante e suportando processos criativos e incerteza. Esta característica se reflete na capacidade de executar modelos de integração incompletos, permitir execução de processos colaborativos (com dependência *start-start*), apoiar a modificação dinâmica do modelo de integração, além de fornecer um formalismo gráfico e executável que permite a modelagem e acompanhamento das dependências entre processos;

- A interoperabilidade nos níveis de modelagem e execução entre PSEEs. É possível definir conexões entre atividades modeladas em diferentes formalismos, e executando em diferentes PSEEs, através da tradução desses formalismos para o formalismo apresentado no meta-modelo do APSEE-Integrate.

O ambiente APSEE possui uma arquitetura ampla e em constante evolução. Estão sendo construídas várias ferramentas de gerência de processos para serem integradas na arquitetura. Trabalhos do mesmo grupo estão

desenvolvendo propostas para adaptação de processos de software e utilização do APSEE para processos de educação a distância, dentre outros trabalhos.

A solução proposta pelo APSEE-Integrate é um passo para flexibilizar e aumentar o nível de automação fornecido por PSEEs, podendo ser útil para incrementar a qualidade dos produtos e diminuir o seu tempo de desenvolvimento. O APSEE-Integrate permite que diferentes tecnologias de suporte a processos de software sejam integradas, aproveitando as vantagens inerentes a cada PSEE.

AGRADECIMENTOS

Ao CNPq (Conselho Nacional de Pesquisa) pelo apoio financeiro a este trabalho.

REFERÊNCIAS

[1] BARDOHL, R. et al. "Appliccation of Graph Transformation to Visual Languages". In: Ehrig, H.; Engels, G.; Kreowski, H-J.; Rozenberg, G. (Eds.) *Handbook of Graph Grammars and Computing by Graph Transformation: Applications, Languages and Tools*. Volume 2. World Scientific, Singapore, 1999.

[2] BARDOHL, R. "GenGED - Visual Definition of Visual Languages based on Algebraic Graph Transformation". PhD Thesis. Technische Universität Berlin. Kovac Verlag, Hamburg, 2000.

[3] BARTHELMESS, P. "Collaboration and coordination in process-centered software development environments: a review of the literature," *Information and Software Technology*, vol. 45, nº 13, pp. 911-928, 2003.

[4] BEN-SHAUL, I.; KAISER, G. "A paradigm for descentralized process modelling and its realization in the Oz environment," in *Proc. 1996 16th International Conference on Software Engineering*, pp. 179-188.

[5] CUGOLA, G.; GHEZZI, C. "Software Processes: a Retrospective and a Path to the Future". In: *Fifth International Conference on Software Process*, Lisle. Proceedings... Jun 1998.

[6] CUGOLA, G.; GHEZZI, C. "Design and implementation of PROSYT: a distributed process support system," in *Proc. 1999 IEEE 8th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pp. 32-40.

[7] CUGOLA, G. et al. "Support for software federations: the PIE platform", in *Proc. 2000 European Workshop on Software Process Technology.*

[8] ESTUBLIER, J.; AMIOUR, M.; DAMI, D. "Building a federation of process support systems," in *Proc. 1999 Siplan, Sogmod, Sigsoft WACC Work, Activity, Coordination and Cooperation Conf.*, pp. 22-26.

[9] GRUHN, V. "Process-centered software engineering environments: a brief history and future chalenges," *Anals of Software Engineering*, vol. 14, pp. 363-382, 2002.

[10] GRUHN, V.; WELLEN, U. "Process landscaping: modelling distributed processes and proving properties of distributed distributed process models," *Lecture Notes in Computer Science: Unifying Petry Nets – Advance in Petry Nets*, vol. 2128, 2001.

[11] GRUNDY, J. C. et al. "A decentralized architecture for process modelling and and enactment, " *IEEE Internet Computing*, pp. 53-62, 1998.

[12] REIS, C. A. L. "Uma abordagem flexível para execução de processos de software," Tese de doutorado, Instituto de Informática, Universidade Federal do Rio Grande do Sul, 2003.

[13] WANG, A. "Support for mobile software processes in CAGIS" in *Proc. 2000 European Workshop on Software Process Technology*.

[14] WANG, A. "A process centred environment for cooperative software engineering, " in *Proc. 2002 SEKE.*

Alternativa de Infraestructura de Clave Pública Basada en el uso de DNSSEC

Rolando Chaparro, Pablo Greenwood, Benjamín Barán
Universidad Nacional de Asunción, Centro Nacional de Computación
Asunción, Paraguay, CC1439
{rfox, pgreen, bbaran}@cnc.una.py

Resumen

El modelo de PKI más ampliamente difundido se basa en el uso de certificados digitales emitidos por Autoridades de Certificación o CAs (*Certificate Authorities*). Por lo general, las CAs están desvinculadas de la infraestructura de red sobre la que se necesita validar y utilizar los certificados. Esta disociación presupone algunas importantes limitaciones. En este artículo se define un modelo de PKI en el que las aplicaciones, en lugar de recurrir a las tradicionales CAs, utilizan las extensiones de seguridad del DNS, conocidas como DNSSEC, como base para la provisión de los servicios fundamentales de seguridad.

Palabras clave: Redes, Seguridad de Datos y Criptografía, PKI, DNS, DNSSEC.

Abstract

The most widely spread PKI model is based on digital certificates issued by Certificate Authorities (CA). In general, there is not a strong connection between these CAs and the underlying network infrastructure on which certificates must be validated and used. This dissociation entails an appreciable number of constraints. This paper proposes an alternative PKI model where applications take advantage of DNS security extensions (know as DNSSEC) as a foundation to build security services.

Keywords: Networking, Data Security and Cryptography, PKI, DNS, DNSSEC.

1 Introducción

Una PKI (*Public Key Infraestructure*) es el medio que facilita el acceso a las claves públicas, asegurando a sus usuarios la correspondencia unívoca de las mismas con sus respectivos propietarios. Este concepto se ha establecido en los últimos años y se ha usado como punto de referencia para proyectar una nueva generación de aplicaciones que hagan posible la provisión de servicios de seguridad a gran escala, sobre todo en sistemas abiertos como Internet. Sin embargo, las actuales implementaciones de PKI presentan aún estimables desafíos, lo cual ha dado origen a diversos estudios, propuestas y trabajos de investigación.

La mayoría de los modelos de PKI se basan en el uso de certificados digitales [30,31], los cuales comúnmente utilizan el formato X.509 [20]. Un certificado es en esencia un vínculo entre una clave pública y los atributos que identifican a una determinada entidad. En el modelo de certificación más difundido, este vínculo es mantenido por organizaciones tipo TTP (*Trusted Third Party*)[1], las que por lo general adoptan la forma de Autoridades de Certificación o CAs (*Certificate Authorities*).

Simultáneamente al auge de las PKIs, en los últimos años se ha venido trabajando en el marco de la IETF[2] en el desarrollo de extensiones de seguridad para el Sistema DNS de Internet [1,27]. Estas extensiones, conocidas como DNSSEC [3,9], han sido diseñadas con el fin de proporcionar servicios de autenticación del origen de los datos del DNS mediante verificaciones criptográficas.

DNSSEC posee algunas características significativas que serán analizadas en este documento con el propósito de demostrar que puede contribuir a definir una alternativa de PKI con un enfoque diferente. Este enfoque se basa en el principio de que los servicios de seguridad deben sustentarse en la propia infraestructura de red y en sus organizaciones, en lugar de depender de agentes externos.

En la sección 2 de este artículo se hace una muy breve presentación de DNSSEC. En la sección 3 se resumen las características elementales de las PKIs de Internet basadas en el concepto de TTPs y en la sección 4 se señalan algunas limitaciones importantes de las mismas. En la sección 5 se propone un nuevo modelo en el que las aplicaciones utilizan a DNSSEC como base para la construcción de servicios fundamentales de seguridad. Finalmente, en la sección 6 se ofrecen las conclusiones de este trabajo.

1 *Trusted Third Party:* tercera parte en la que los usuarios deben depositar su confianza
2 IETF: *Internet Engineering Task Force*

2 Fundamentos de DNSSEC

Las extensiones DNSSEC proporcionan servicios de autenticación del origen de los datos del DNS mediante verificaciones criptográficas[3]. Para ello se realizan firmas digitales sobre los registros de la base de datos del DNS. Estas firmas y sus claves públicas asociadas son insertadas en las tablas de zona [28] mediante registros tipo DNSSIG y DNSKEY respectivamente, los cuales son incluidos y enviados en los mensajes de respuesta a las consultas DNS [4]. Los clientes, denominados *resolvers*, finalmente verifican los datos de respuesta empleando las firmas digitales y las claves públicas recibidas [2].

La legitimidad de las claves públicas es corroborada mediante el recorrido del *chain of trust* del DNS [14]. En este modelo de autenticación, las claves públicas de una zona son convalidadas por su zona padre a través de una firma digital denominada DS (*Delegation Signing*) [14]. Esta relación entre una zona y su antecesora se reproduce siguiendo la jerarquía del árbol de DNS hasta llegar a una zona cuyas claves públicas son consideradas confiables (*secure entry keys*).

En resumen, DNSSEC provee una infraestructura que soporta el uso de claves públicas para dar servicios de seguridad, cuenta con un mecanismo de distribución de estas claves (el DNS) y confiere autenticación a las mismas. Aunque estas características se ajustan al concepto de PKI, se debe notar que DNSSEC ha sido ideado únicamente para brindar protección contra los ataques de tipo *spoofing* al DNS [6]. DNSSEC se diferencia además de una PKI tradicional en que el protocolo no utiliza certificados, el directorio es descentralizado y la autenticación se basa en autoridades subdivididas y delegadas [3].

3 Las Infraestructuras de Clave Pública en Internet

De todos los modelos de PKI, el basado en certificados digitales emitidos por TTPs, es el que ha logrado mayor difusión en la Internet global, así como en soluciones corporativas, llegando de forma masiva a una gran cantidad de usuarios. Por consiguiente, de aquí en adelante este artículo se referirá principalmente a este paradigma de PKI, cuyas características elementales se resumen a continuación.

A partir del concepto de certificación de TTPs han sido diseñadas varias propuestas de PKI, pero el esquema adoptado por X.509 es actualmente la base de casi todos los productos existentes. Conforme se ilustra en la Figura 1, el rol central es desempeñado por las CAs, que emiten los certificados a las entidades. Opcionalmente las CAs pueden delegar labores administrativas a otras organizaciones denominadas RAs (*Registration Authorities*), pero son las CAs las que en realidad actúan como TTPs, en cuyos certificados los usuarios deben depositar su confianza. El estándar X.509 ha evolucionado en sucesivas versiones hasta la actual X.509v3 [21], cuyo formato de certificado se aprecia en la Figura 2.

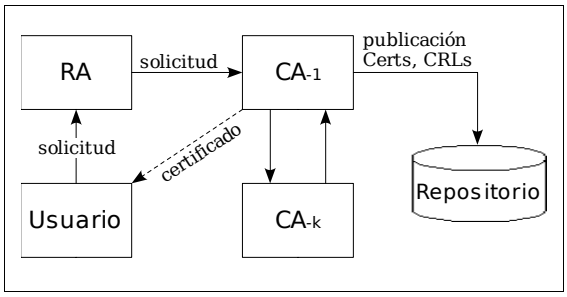


Figura 1: Estructura básica de PKI tipo TTP

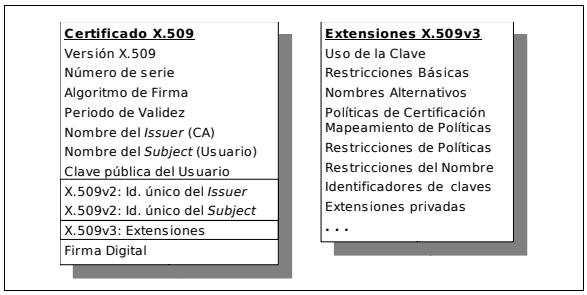


Figura 2: Formato del certificado X.509v3

El requerimiento fundamental de un certificado es establecer y mantener de forma precisa el vínculo entre los atributos que lo constituyen. En la mayoría de los casos, los atributos de interés dominante son una clave pública y los datos que identifican al propietario de esta clave.

X.509 fue diseñado como un esquema de certificación que define una infraestructura de clave pública para autenticar los servicios del directorio X.500 [18]. Los certificados y otras estructuras de datos de la PKI son guardados en el mismo directorio. La identificación de un certificado y su consecuente ubicación en el directorio X.500 se realiza en base a un nombre distintivo o DN (*distinguished name*), que describe un camino jerárquico único en el directorio, entre la raíz y el propietario del certificado [19]. Algunos de los atributos más importantes de los certificados X.509, como el *Subject Name* y el *Issuer Name*, están en formato DN. Un DN consta de un conjunto de atributos abreviados, cada uno asociado a un nivel en la jerarquía X.500. Ejemplo: {C=PY, O=UNA, OU=CNC, CN=Rolando Chaparro Fox}[4].

3 Actualmente se está concluyendo la revisión y corrección de las especificaciones originales del RFC-2535 (1999)
4 C (*country*), O (*organization*), OU (*organizational unit*) y CN (*common name*)

X.509 define un método de revocación de certificados, denominado CRL (*Certificate Revocation List*), consistente en una estructura de datos firmada por la CA y actualizada periódicamente, la cual contiene la lista de certificados revocados. Recientemente han sido desarrollados protocolos como OCSP (*Online Certificate State Protocol*) que permiten realizar consultas en línea del estado de los certificados [26].

4 Algunas Limitaciones de las PKIs Basadas en las Actuales CAs

Es posible diferenciar a las CAs en función de su tipo de organización y a la manera en que están facultadas a emitir certificados. A continuación se presenta una clasificación tentativa[5] basada en [24]:

- *Organizacional*: Típicamente es una universidad o empresa que actúa como su propia CA para emitir certificados a las sub-unidades que la componen y a los integrantes de las mismas.
- *Geopolítica*: El gobierno, o alguna organización que reciba la anuencia de éste, expide los certificados a las entidades en base a un modelo nacional de PKI.
- *Universal*: La autoridad de certificación de una hipotética PKI global que es reconocida por todos.
- *Propietaria*: Una organización que controla un determinado espacio de nombres y que emite certificados que son utilizados en ese contexto.
- *Comercial*: Empresa que actúa como TTP y cuyo negocio es la emisión de certificados en los que las entidades de la PKI deben depositar su confianza.

Una inmensa mayoría de los certificados de sitios de Internet son emitidos por CAs que corresponden a este último grupo. Tomándolo como referencia, los puntos a continuación están orientados a señalar algunos desafíos que enfrentan estas PKIs, en busca de una mejor alternativa desde el punto de vista técnico, con mayores ventajas para los usuarios.

4.1 Relación de confianza desde la perspectiva del usuario.

Es generalmente aceptada la idea de que el usuario es responsable de una importante parte de las evaluaciones de confianza en el proceso de autenticación. Por ejemplo, las denominadas *trusted-CAs* son hipotéticamente aquellas CAs elegidas por el usuario para depositar su confianza. Sin embargo, en la práctica adquieren este rango por el sólo hecho de que sus certificados auto-firmados (certificados raíz) están pre-instalados en el *browser*. Es decir, el usuario promedio no se toma el trabajo de examinar los certificados raíz de las CAs predefinidas para decidir si confía o no en ellas.

De forma similar, se sostiene que los certificados recibidos contienen importantes datos sobre los que se espera que el usuario realice algún tipo de discernimiento o juicio de valor. Este es el caso de los nombres del propietario (*Subject*) y de la CA (*Issuer*). Pero aquí nuevamente se presentan algunas inconsistencias.

Veamos primero el caso del *Issuer*. Basta que los certificados recibidos estén firmados por alguno de los muchos certificados raíz incluidos en el *browser*, para que haya una aceptación implícita de los mismos[6]; pero acabamos de mencionar que las *trusted-CAs* nunca fueron validadas por el usuario. Esto implica que, en la práctica, los vendedores de las aplicaciones intervienen de forma decisiva condicionando los criterios de confiabilidad del usuario. Esta situación de hecho es una deformación del concepto de confianza.

La situación no es muy diferente respecto al *Subject*. Supongamos que existen dos empresas con la misma denominación, digamos "*Multimedia Inc*"; una es una agencia de publicidad y la otra desarrolla software multimedia[7]. Al acceder al sitio web *www.multimedia.com.py*, el usuario difícilmente podrá discernir a partir de los datos incluidos en el *Subject*, a cual de las dos empresas se refiere el certificado recibido (si es que acaso este usuario es de los pocos que optan por examinar certificados y además, coincidentemente, sabe de la existencia de otra empresa con el mismo nombre). Este es en el fondo un problema inherente a la multiplicidad y a la diversidad de los espacios de nombres, que ninguna de las soluciones actuales puede resolver con absoluta precisión [11].

Finalmente, dado que la mayoría de los usuarios ni siquiera revisa el contenido de los certificados recibidos, ni de los certificados raíz, tiene poca importancia que sus atributos sean datos estructurados y legibles por las personas. En la práctica, el usuario se vale de otras señales que le ayudan a identificar a la empresa una vez que haya accedido al sitio web; o bien, tiene algún grado de certeza de que el *host* corresponde a la organización a la que él tiene interés de contactar.

5 En esta clasificación, los tipos de CA no son mutuamente excluyentes
6 Este es el comportamiento por defecto del explorador web utilizado por cerca del 90% de los usuarios de Internet
7 Al tratarse de diferentes rubros, este tipo de igualdad de nombres es válida en muchas legislaciones del mundo

4.2 Manejabilidad de las soluciones X.509

Dado que casi la totalidad de los vendedores se basan en X.509, puede tenerse la sensación de que este escenario siempre resulta conveniente a todas las partes. Después de todo, para eso están los estándares. Pero el estándar X.509 es muy general y por lo tanto tiene varias imprecisiones. Esto implica que en la práctica se necesita definir un perfil específico para implementar un sistema real de certificación.

Hoy existen múltiples vendedores de PKI que emplean diferentes perfiles de X.509. Como consecuencia, las soluciones difieren en sus implementaciones, incluso en aquellos atributos más importantes de los certificados. Esta multiplicidad de soluciones implica también entornos de operación divergentes, cada uno con sus propias políticas, herramientas y prácticas. En muchos casos se vuelve necesario realizar trabajos personalizados de adecuación para lograr un aceptable grado de interoperabilidad.

Actualmente se están realizando esfuerzos para minimizar los inconvenientes de interoperabilidad, desde diferentes áreas y en diferentes etapas de definición. PKIX [17] es tal vez el más notable de ellos. Sin embargo, no puede perderse de vista que la idea de interoperabilidad comúnmente implica transitividad de la confianza depositada en las CAs y en sus sistemas de certificación. Esto a pesar de que la confianza es generalmente un concepto relativo y no transitivo.

Muchas aplicaciones actuales basadas en la tecnología de clave pública podrían beneficiarse de soluciones más simples, con menor grado de dependencia de las complejidades derivadas del uso de X.509 y de las relaciones de confianza.

4.3 Estructura de costos

Idealmente, la escala de precios de cualquier producto debería estar definida en función y en proporción a los servicios y beneficios obtenidos por los consumidores. En este sentido, la actual estructura de costos de las CAs define niveles de seguridad muy generales para los certificados. Difícilmente estos niveles puedan corresponder de forma adecuada a los requerimientos de todo el rango de aplicaciones en las que hoy son empleados los certificados.

Por otra parte, las entidades deben realizar una considerable inversión si pretenden tener varios certificados compatibles con las rutas de certificación pre-instaladas en las aplicaciones de uso masivo[8]. Por ejemplo, al momento de escribir este artículo, VeriSign[9] estaba ofertando un paquete de 100 (cien) certificados a un precio de US$ 57.000[10]. Si bien a las entidades que adquieren este paquete se les facilita herramientas para administrar sus certificados, los mismos en realidad son firmados por la CA.

A esto hay que agregar que, al estar pagando en esencia por la firma de sus certificados, una entidad que decida cambiar de CA debe realizar una reinversión completa. Esto es así aún cuando sólo haya usado el certificado por un breve tiempo, o no lo haya usado en absoluto. Es posible que a partir de modelos alternativos de PKI, surjan también nuevos modelos de negocios; aunque debe reconocerse que muchas veces es más rentable desarrollar productos a partir de estructuras y tecnologías ya establecidas.

4.4 Repositorios y servicios de directorio

DAP (*Directory Access Protocol*) es el protocolo original de acceso al servicio de directorio X.500. El mismo fue diseñado para soportar la infraestructura de clave pública definida por X.509. Sin embargo, LDAP(*Lightweight Directory Access Protocol*), que es la herramienta utilizada actualmente por la mayoría de las implementaciones de PKI, ha tenido un origen distinto y con diferentes motivaciones [32]. Esto ha originado una serie de desajustes operacionales, los cuales se describen de forma precisa en [8].

Otro aspecto que aún necesita de una solución satisfactoria es que los repositorios de certificados puedan ser desplegados de forma masiva en Internet, brindando mecanismos simples de recuperación. En este sentido, trabajos recientes han explorado la posibilidad de emplear al sistema DNS como repositorio de certificados de una PKI [22].

De forma similar, las propias CAs han planteado utilizar al DNS para facilitar a los usuarios la localización de sus repositorios de certificados y listas de revocación. La propuesta denominada RLS (*Repository Locator Service*) es actualmente un *draft* de la IETF que plantea asociar a un nombre de dominio, el protocolo y la dirección de estos servicios de la PKI [7].

8 Navegadores web o el software de correo electrónico
9 VeriSign es actualmente una de las empresas más importantes en el mercado de las CAs comerciales
10 Corresponde a los certificados con soporte de cifrado simétrico de 128 bits

4.5 Nombres e identidad

El nombre de la entidad propietaria es uno de los atributos decisivos y de mayor importancia en un certificado. El propio estándar X.509 establece que es responsabilidad de una CA asegurar la unicidad de los nombres de las entidades para las que emite los certificados.

El diseño original de X.509 presupone que los nombres en el certificado están en formato DN[11]. Para garantizar su unicidad, los DNs están organizados de forma jerárquica y se asume que hay una autoridad que hace que los mismos sean globalmente únicos y significativos. Sin embargo, esta idea nunca se puso en práctica. En consecuencia, al no estar integrados los servicios de directorios, ni estandarizados los criterios de asignación de nombres, las CAs no cumplen con este requisito de unicidad.

Dos organizaciones distintas podrían tener el mismo DN en dos CAs diferentes, también es posible que una misma organización tenga DNs disímiles en ambas CAs. Esta ambigüedad representa un potencial problema de interoperabilidad. Incluso en el ámbito cerrado de una misma CA, los DNs de sus clientes no reflejan verdaderamente ninguna organización jerárquica. El siguiente ejemplo ilustra una de las tantas discordancias que podrían ocurrir a raíz de la permisividad con la que se manejan los DNs en los certificados emitidos por las CAs:

{C=PY, O=Gobierno Central, OU=Secretaría de Economía, ...}
{C=PY, O=Gobierno Nacional, OU=Secretaría de Salud Pública, ...}
{C=PY, O=Poder Ejecutivo, OU=Secretaría de Educación y Cultura, ...}

Como puede notarse, la nomenclatura jerárquica implícita en los DNs pierde significación. En la práctica, las aplicaciones procesan los nombres en los certificados como si fueran simples conjuntos de datos no estructurados. Además, como se ha mencionado en la sección 4.1, tiene escasa relevancia que los mismos sean legibles por una persona, dado que el usuario promedio no examina los certificados.

La situación del ejemplo arriba señalado no se reproduce tan fácilmente en el Sistema DNS de Internet, donde la operación del servicio y la responsabilidad de asignación de los nombres de dominio se distribuye y delega efectivamente de manera jerárquica. El DNS es una verdadera estructura de árbol en la que se asegura la unicidad de los nombres.

El DNS se encuentra en un ámbito externo, desvinculado de las PKIs. A pesar de ello, existe una clara dependencia por parte de las CAs hacia el DNS, dado que éste es el espacio de nombres de la infraestructura de red en la que se pretende desplegar los servicios de seguridad. Esta dependencia se vuelve evidente en servicios abiertos como SSL (*Secure Socke Layer*) [12]. Para hacer uso efectivo de este protocolo, los certificados de sitios web expedidos por las CAs incluyen en el DN del *Subject* el nombre DNS del sitio. Sólo así es posible identificar en el ámbito de Internet al propietario de un certificado.

4.6 Confianza y autoridad

Las CAs son terceras partes en las que los usuarios deben depositar su confianza. Sin embargo, en el caso de las actuales PKIs de uso masivo, se ha visto en la sección 4.1 que esta relación en la práctica no se establece en base a verdaderos criterios de confiabilidad. Aunque así fuera, la confianza es en realidad una relación compleja, no transitiva, de carácter relativo, no cuantificable y culturalmente influenciada.

Sin embargo, la autenticación no necesita estar basada en la confianza. Si una entidad es autoritativa[12] sobre un determinado espacio de nombres, la débil y difusa noción de confiabilidad es irrelevante. Uno no se pregunta si se deposita o no confianza en una determinada empresa para entregar carnés de identificación a sus propios empleados, o en un determinado país para expedir pasaportes a sus ciudadanos. En estos ejemplos, cada entidad emisora es autoritativa sobre el espacio de nombres en el que estas credenciales son emitidas, de modo que la confianza es intrínseca [24].

Se sostiene que los usuarios deben confiar en las CAs, pero son muy limitados el aval y la responsabilidad que éstas pueden ofrecer sobre la autenticidad, la integridad o la exactitud de los datos contenidos en los certificados, lo cual se refleja en sus términos contractuales [13]. Esto se debe sobre todo a que las CAs no son autoritativas sobre ningún espacio de nombres, y esto incluye al espacio de nombres de la infraestructura de red en la que los certificados son efectivamente evaluados y utilizados (el Sistema DNS de Internet). En cambio, las entidades autoritativas de un espacio de nombres tienen una responsabilidad igualmente intrínseca. Ellas no pueden eludir las obligaciones que se desprenden de su autoridad.

11 Si bien X.509v3 permite otros tipos de nombres, la nomenclatura DN es mantenida por todas las CAs

12 Para la mayoría de los tipos de nombres existen entidades del mundo físico que tienen el control o que manejan el espacio en el cual los nombres son asignados. Se dice de estas entidades que son autoritativas (que tienen autoridad) sobre esos espacios de nombres.

4.7 Mecanismos de revocación

Las listas de revocación de las PKIs son escasamente utilizadas en la práctica. Por ejemplo, SSL, que es la principal aplicación de seguridad de la actualidad, no descarga los CRLs. Esto se debe en parte al gran tamaño de algunos de ellos y a la potencial saturación del servicio, el cual es esencialmente centralizado.

Ante una situación de clave comprometida, las CAs deben desplegar procedimientos adicionales de seguridad que pueden demorar, aumentando la latencia entre el reporte de revocación y la distribución efectiva de la información a todas las partes.

Los protocolos de verificación en línea también tienen sus limitaciones. Si bien OCSP puede reportar el estado de un certificado, en la práctica este método aumenta la complejidad del modelo al imponer un nuevo requerimiento de seguridad. El cliente debe también verificar la integridad y la autenticidad del origen de los datos de este nuevo servicio de validación.

5 Infraestructura de Clave Pública Basada en DNSSEC

Un certificado digital es en esencia un vínculo entre una clave pública y los atributos que identifican a una determinada entidad. El modelo de certificación más difundido plantea que este vínculo es mantenido por CAs comerciales, en las cuales los usuarios deben depositar su confianza. Sin embargo, en la sección 4 se ha visto que este enfoque trae consigo una serie de limitaciones[13].

En esta sección se plantea una alternativa de PKI desde una perspectiva diferente que no depende de agentes externos. Se propone un nuevo modelo en el que la propia infraestructura de red de Internet contribuye a la provisión de los servicios fundamentales de seguridad. Para ello se vale de la jerarquía de autenticación implícita en el DNS y sus extensiones de seguridad.

5.1 Aspectos generales del modelo

Además de la información estrictamente necesaria para realizar la resolución de nombres, el diseño original del DNS faculta a una organización a incluir en su base de datos otros atributos[14] asociados a sus entidades (*hosts*, sub-dominios, personas, roles), permitiendo a las aplicaciones hacer uso de los mismos.

Basado en este mismo principio, el modelo aquí expuesto propone la creación de un RR destinado a publicar a través del DNS las claves públicas asociadas a una entidad. Este nuevo registro, denominado APPKEY, permitirá a las aplicaciones localizar, recuperar y hacer uso de estas claves. Dada su comprobada eficacia como servicio de directorio simple, su escalabilidad, su naturaleza distribuida y su uso extendido en toda la red Internet, el sistema DNS resulta ideal para este trabajo.

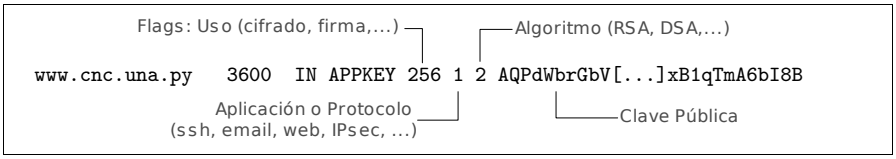


Figura 3: Ejemplo de registro APPKEY con su posible conformación de RDATA

La integridad del registro APPKEY y la legitimidad de su origen, estarán dadas por DNSSEC y su *chain-of-trust*. La posibilidad de obtener las claves públicas ya autenticadas representa muy claramente una alternativa a la utilización de certificados emitidos por CAs externas.

Una aplicación que ha recuperado una clave pública a través de DNS/DNSSEC puede pasar directamente a comprobar la autenticidad de la otra parte con la que pretende establecer comunicación[15]. Igualmente, valiéndose de esta clave pública, puede proceder a negociar y a establecer otros servicios de seguridad como la confidencialidad.

5.2 La infraestructura de red como sustento de los servicios de seguridad

El DNS tiene como finalidad principal facilitar a los usuarios la identificación y localización de entidades, típicamente de *hosts*, mediante el mapeo de nombres de dominio a números IP. Si bien estos dos atributos, Nombre DNS + Dirección IP, eran suficientes para cumplir con este cometido en los inicios de Internet, los actuales requerimientos de seguridad de una red global demandan mayor esfuerzo.

13 Información más completa y detallada puede encontrase en [10,13,29].
14 Mediante *Resource-Records* como MX, SRV, HINFO, TXT o CERT.
15 Para ello, la otra parte debe demostrar la posesión de la correspondiente clave privada.

En este contexto fueron diseñadas las extensiones DNSSEC, las cuales permiten confirmar la asociación Nombre DNS + Dirección IP. No obstante, un intruso puede aún apropiarse indebidamente de un número IP. En tal sentido, este trabajo plantea que una entidad puede ser positivamente identificada a partir de sus atributos Nombre DNS + Dirección IP + Clave Pública, y que este último dato también debería poder obtenerse del servicio de infraestructura DNS. Esta es justamente la finalidad el registro APPKEY.

En Internet, cada organización tiene autoridad sobre sus nombres de dominio. En base a esta autoridad definen los nombres particulares que tendrán sus entidades. La presente propuesta refleja fielmente este importante principio, pues cada organización es libre de firmar y publicar las claves de sus entidades.

Por otra parte, fácilmente se infiere que en este modelo, las organizaciones que administran los nombres de dominio correspondientes a cada país (*ccTLD*[16] *Authorities*) desempeñarán de forma espontánea un rol equivalente al que hoy cumplen las CAs. Esta idea tiene sólidos fundamentos:

- Existe gran similitud entre las operaciones realizadas en el registro de nombres de dominio bajo un ccTLD y las solicitudes de certificados a una CA. Actualmente se realizan de forma redundante trámites que son equiparables, y en la práctica es un requisito tener antes un nombre de dominio.
- Las *ccTLD Authorities* tienen un vínculo más cercano con las empresas y organizaciones nacionales. Esto les brinda mayor autoridad para los procesos de verificación respecto a las CAs comerciales. Incluso hay muchos países en los que las CAs ni siquiera tienen presencia.
- Las CAs recurren a las bases de datos de las *ccTLD Authorities* como práctica normal para validar parte de la información suministrada por sus clientes.
- Las *ccTLD Authorities* son las entidades autoritativas para la delegación de los nombres DNS a las empresas y organizaciones con presencia en Internet en cada país. Considérese que el DNS es el espacio de nombres de la infraestructura de red sobre la que se pretende desplegar los servicios de seguridad.
- Al estar estrechamente vinculadas al funcionamiento de Internet en sus respectivos países, las *ccTLD Authorities* siempre están presentes; pueden cambiar de administración, pero no desaparecer.
- Son organizaciones sin fines comerciales, ampliamente reconocidas y que tienen el respaldo de la Comunidad Internet local de cada país. Por lo general son asociaciones de ISPs y usuarios, o universidades, o comités multi-sectoriales, en ocasiones con participación de agencias gubernamentales.

En resumen, esta propuesta defiende el principio de que los servicios de seguridad deben sustentarse en la propia infraestructura de red y en sus organizaciones, en lugar de depender de agentes externos.

5.3 La autenticación como base del modelo propuesto

En toda infraestructura de red, la autenticación es el servicio de seguridad fundamental, del que dependen y a partir del cual se *construyen* todos los demás servicios, por ejemplo la confidencialidad. En Internet, y en cualquier red TCP/IP, se plantean siempre tres preguntas centrales desde el punto de vista de la autenticación. Para ilustrar cómo responde el modelo propuesto a estas interrogantes, retomemos el ejemplo del usuario tratando de acceder al sitio web de *Multimedia Inc.* Las preguntas en cuestión son:

a. Es en verdad *www.multimedia.com.py* un nombre DNS perteneciente a la organización *Multimedia Inc,* con la que el usuario quiere establecer contacto ?

b. Es la dirección IP obtenida a través del DNS la que corresponde realmente a *www.multimedia.com.py* ?

c. Se está realizando la comunicación con el *host* verdadero o con otro que se ha apropiado indebidamente de su número IP ?

Respecto a la primera pregunta, si bien es posible argumentar que el *Subject* es un dato suficientemente descriptivo, en la sección 4.1 se han mencionado algunas razones de peso que cuestionan el uso real que se da actualmente a este y a otros atributos de los certificados. Se ha visto además que existen espacios de nombres que no garantizan unicidad, y que aún las actuales soluciones basadas en X.509 no responden con absoluta precisión a esta interrogante.

Sin embargo, como se indica en la sección anterior, los administradores de los ccTLDs son tanto o más confiables que las actuales CAs para identificar a las organizaciones en el ámbito de un determinado país. Además, tienen autoridad sobre el espacio de nombres de la infraestructura de red subyacente.

No obstante, hay alternativas para aquellos usuarios en situación de duda. Las aplicaciones podrían por ejemplo disponer de una operación opcional para recuperar, a partir del servicio *whois* de los ccTLDs, los datos requeridos para identificar a una entidad. Las bases de datos *whois* contienen información generalmente más rica y completa que la que puede encontrarse actualmente en un certificado.

16 ccTLD: *Country Code Top Level Domain.* Son los nombres de dominio de los países (.ar, .br, .cl, etc.)

En cuanto a la segunda pregunta, la autenticidad de los datos del DNS estará dada por DNSSEC, eliminando la necesidad de certificados de CAs externas debido a que las claves públicas estarán legitimadas por las propias organizaciones y por la jerarquía de autenticación de DNSSEC.

Finalmente, la tercera pregunta sólo puede responderse demostrando en términos criptográficos la posesión de la correspondiente clave privada. Este es un paso indispensable en cualquier protocolo o aplicación basada en la criptografía de clave pública.

5.4 Obtención y autenticación de las claves públicas

La Figura 4 ilustra de forma simplificada cómo una aplicación cliente, consultando a su servidor DNS local, recupera la dirección IP y la clave pública del *host* destino.

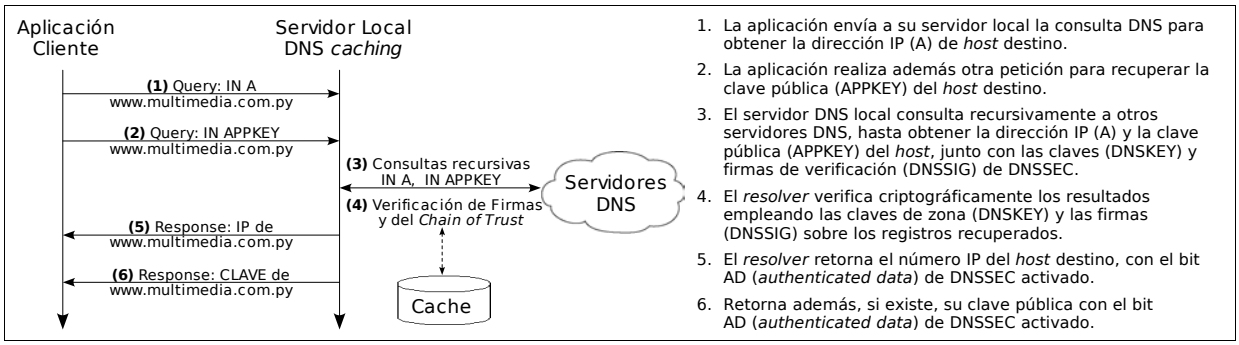


Figura 4: Pasos abreviados de la interacción con DNS/DNSSEC para recuperar una clave pública

Gran parte de la complejidad del protocolo DNSSEC reside en el *resolver*. En la práctica los pasos 3 y 4 de la Figura 4 no se efectúan necesariamente de forma secuencial. Aquí se hace una abstracción respecto a los algoritmos concretos que el *resolver* emplea para obtener los resultados parciales a sus consultas y para realizar las verificaciones. De hecho, estos procedimientos se encuentran aún en estudio [3].

A partir de la obtención y validación de la clave pública del registro APPKEY, la aplicación está ahora en condiciones de autenticar a la entidad destino. Luego podrá intercambiar claves secretas y negociar los algoritmos de cifrado simétrico y de MAC (*message authentication code*) para establecer servicios de seguridad como la integridad de los datos y la privacidad, tal como se hace en protocolos como SSL, TLS.

Está fuera del alcance de este artículo definir los mecanismos específicos para la realización de estas operaciones. Sin embargo, en la siguiente sección se mencionan algunas posibles alternativas con el fin de realizar estimaciones teóricas de rendimiento.

5.5 Algunas consideraciones acerca del rendimiento

Con el fin de facilitar su apreciación, en esta sección se ha elegido como ejemplo el acceso a un servidor web/SSL para realizar comparaciones de referencia. Sin embargo, el modelo propuesto puede también ajustarse a otros tipos de aplicaciones, como el intercambio seguro de mensajes de correo electrónico.

5.5.1 Tamaño y cantidad de paquetes

Al realizar una primera comparación, se observa en la Figura 5 que el servidor SSL envía su clave pública en el certificado X.509, el cual es transferido en un único mensaje del sub-protocolo *Handshake.* Sin embargo, en las Tablas 1 y 2 se aprecia que ambos enfoques emplean en realidad un *round-trip* completo para la entrega de la clave pública[17]. Por un lado, el mensaje *Certificate* de SSL y su necesario TCP *Acknowledge*; y por el otro, los mensajes *Query* y *Response* de DNS[18] (pasos 2 y 6 de la Figura 4).

Sin embargo, es factible que la aplicación cliente obtenga con una sola consulta DNS la dirección IP y la clave pública. Para ello, este último dato puede incluirse en la sección adicional del mensaje de respuesta a la consulta DNS tipo A[19]. Esto eliminaría los mensajes vinculados a la petición explícita de APPKEY, aunque aún podría realizarse de ser necesario. Es decir, el *round-trip* para obtener la clave pública puede reducirse a cero, dado que de todas formas hay que recuperar del DNS la dirección IP.

17 En la Tabla 2, el tamaño estimado de *Query* está basado en el tamaño de un típico mensaje DNS de 800 bits [1]. Para el mensaje *Response* se asume el peor escenario y se utiliza el tamaño máximo del *payload.*

18 Se consideran solo los mensajes intercambiados entre la aplicación y el servidor DNS local para obtener la clave pública. No se toman en cuenta los mensajes recursivos originados por este último.

19 Esta misma idea se encuentra en la sección 3.5 de la especificación original de DNSSEC [9].

Mensaje	*Encabezado TCP*	*Encabezado HS SSL*	*Payload (X.509)*	*Total*
Certificate	20	4	1160	1184
(TCP ACK)	20	-	-	20
Total				**1204**

Tabla 1: Cantidad aproximada de bytes para la obtención del Certificado X.509 en SSL v3.0

Mensaje	*Encabezado UDP*	*Tamaño Estimado de Mensaje DNS*	*Total*
Query	8	100	108
Response	8	512	520
Total			**628**

Tabla 2: Cantidad aproximada de bytes para la obtención de la clave pública en el modelo propuesto

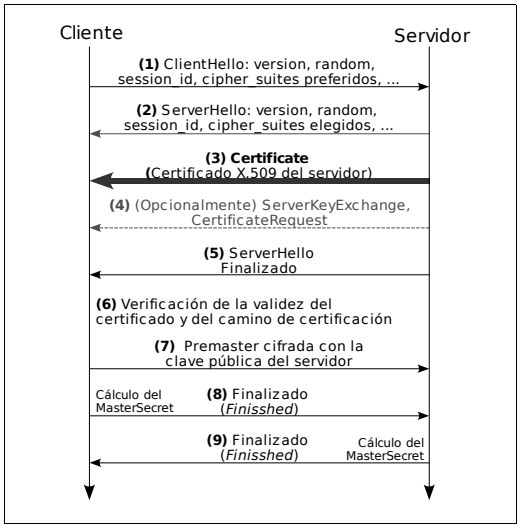


Figura 5: HandShake abreviado de SSL

También en las Tablas 1 y 2 puede verse que la cantidad teórica[20] de bytes enviados y recibidos para obtener la clave pública es mayor en SSL. Esto se debe al tamaño del certificado[21] y al mayor *overhead* de TCP (usado por SSL) respecto a UDP. Este último es el servicio de transporte por defecto de DNS.

No obstante, los valores de la Tabla 2 pueden verse aumentados si se contemplan los mensajes DNS recursivos disparados por el servidor DNS local (paso 3 de la Figura 4). La cantidad de consultas recursivas que lleguen a hacerse depende de varios factores, entre ellos la información ya guardada en el *cache* del servidor local y en los *caches* de los servidores consultados por este.

Esto significa que el *cache* del DNS desempeña un papel muy importante en el modelo propuesto, en particular desde el punto de vista de las cantidades totales de paquetes y bytes transmitidos. El *cache* permite que las consultas recursivas sean realizadas únicamente cuando el TTL del RR APPKEY haya expirado. El resto del tiempo, la clave estará disponible para las aplicaciones en el *cache* del servidor DNS local.

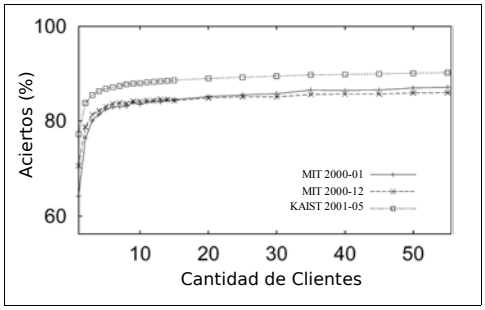


Figura 6: Porcentaje de Aciertos de los *caches* de varios servidores DNS en función a la cantidad de clientes

La Figura 6 muestra uno de los resultados de un reciente trabajo de análisis de la efectividad del *cache* del DNS [23]. En ella se observa que el índice de aciertos del *cache* mejora al aumentar la cantidad de clientes que están consultando, y que los mayores beneficios se notan ya con los primeros 10 a 20 usuarios. A partir de la misma gráfica se puede presumir que, en general, la probabilidad de encontrar un registro en el *cache* de un servidor local aumenta en proporción a la cantidad de consultas que este recibe. Por lo tanto, la mayor parte del tiempo, aquellas claves más requeridas serán recuperadas localmente. Esto representa un mejor aprovechamiento del ancho de banda y es una importante ventaja respecto a SSL.

La inclusión del APPKEY en la sección adicional de un mensaje de respuesta puede hacer que se exceda el tamaño máximo del *payload* de DNS (512 bytes). En tal caso, se emplea el mecanismo de extensión EDNS0 para no realizar un *fall-back* a TCP, como se define en [15]. Además, el tamaño máximo de paquetes UDP en IPv6 es cercano a los 1500 bytes, lo que casi elimina la posibilidad de *overflow*.

Existe también una trabajo denominado SK-DNSSEC [5] que sugiere el uso de criptografía simétrica en DNSSEC. Dicha propuesta plantea una reducción del tamaño de los mensajes, entre otras ventajas.

Además, cabe destacar que, cuando DNSSEC sea parte de la infraestructura de red, el *chain-of-trust* será indefectiblemente verificado para establecer cualquier comunicación TCP/IP normal, por lo que la presente propuesta prácticamente no agregaría *overhead* adicional. En cambio, las aplicaciones que en ese entonces sigan usando certificados de las CAs externas estarían realizando operaciones de forma redundante y a la vez desaprovechando los recursos disponibles.

20 Se omiten las capas inferiores y se presume que no ocurre fragmentación a causa de paquetes muy grandes.
21 Tamaño habitual de un certificado de VeriSign con clave RSA de 1024 bits

5.5.2 Procedimiento abreviado de negociación

En la Figura 5 se observa que en SSL el cliente envía primero la lista de los algoritmos de cifrado y de control de integridad que soporta, luego el servidor responde indicando sus elecciones.

En el modelo propuesto es posible disminuir el *round-trip* de esta fase inicial de negociación. En el mismo RR APPKEY podría incluirse las preferencias del servidor respecto a estos algoritmos. Al obtener con una sola consulta DNS la clave y los algoritmos preferidos del servidor, el cliente podría pre-ajustar sus parámetros y enviar en un único mensaje los datos incluidos en los paso 1 y 7 de la Figura 5. Si el cliente no soporta los algoritmos indicados por el servidor, entonces estas operaciones de *handshake* pueden realizarse de la manera habitual.

5.5.3 Carga computacional de las verificaciones criptográficas

En el modelo propuesto, la carga computacional de la verificación de la clave pública se transfiere de la aplicación del cliente al *resolver* del servidor DNS local. Por ello es importante estimar la sobrecarga potencial que este componente de software podría experimentar.

Además de la verificación de la firma sobre su RRset[22], un simple registro como A o APPKEY requiere del recorrido del *chain-of-trust* para verificar todas las delegaciones. En las nuevas especificaciones de DNSSEC, la carga computacional de las verificaciones para consultas sobre un determinado nivel de dominio n, están dadas por $C=f(n)=2n+1$ [14].

En el modelo propuesto, la consulta DNS para APPKEY se hace de forma casi simultánea a la consulta de la dirección IP. Esto hace improbable que la verificación de la delegación (el primer término en la función anterior) se realice dos veces. Entonces, si se conoce la proporción de sitios web seguros, puede calcularse la sobrecarga computacional SC, respecto a la carga base C:

$$SC=f(n,p)=\frac{1}{2n+1}p$$ *Donde* p es la proporción de sitios web seguros para ese nivel

Estadísticas recientes[23] muestran que 1.53% es la proporción de sitios web seguros en Internet. Asumiendo que aproximadamente esa misma proporción se mantiene para todos los dominios de tercer nivel, la sobrecarga computacional SC es apenas de 0.2% para ese nivel.

Además, aquí el *cache* juega nuevamente un importante papel. La verificación criptográfica se realiza sólo si ha transcurrido el TTL de APPKEY en el servidor DNS local. Esto se aplica para todos los usuarios que accedan a su *cache*. En consecuencia se logra un mejor desempeño computacional colectivo.

5.5.4 Otros recursos computacionales

La inclusión de APPKEY tiene escasa incidencia en los recursos de cómputo del proceso de firma de las zonas. La mencionada proporción de 1.53% de servidores web seguros, es abismalmente pequeña en relación al 98.47% restante que, de igual manera, debe ser firmado conforme lo requiere DNSSEC.

Por el mismo motivo, los servidores DNS, sus *caches* y sus requerimientos de almacenamiento y de memoria RAM tampoco tendrían que verse en general sensiblemente afectados. Debe tenerse en cuenta que las zonas *super gigantes* como .com y .net sólo alojan datos de delegación.

5.6 Cambio y revocación de las claves

Ante una situación de clave comprometida, las CAs externas deben desplegar procedimientos adicionales de seguridad que pueden demorar, aumentando la latencia entre el reporte de revocación y la distribución de la información a todas las partes. En el modelo propuesto, en cambio, es más sencilla y natural esta verificación, así como la comprobación del origen de la nueva clave, dado que estas operaciones se realizan en el ámbito interno de la misma organización.

Los mecanismos de remoción de una clave y de publicación y distribución segura de la nueva clave a los usuarios, están implícitos en el DNS y en las extensiones DNSSEC, y no dependen de la intervención de una tercera parte. Además, la frecuencia de renovación de las claves y su tiempo de vida en los *caches* son controlados por la propia organización mediante la actualización de los datos de zona y la definición de TTLs, sin depender de políticas particulares de las CAs.

22 *RRset*: Conjunto de *resource-records* con el mismo nombre, clase y tipo
23 Fuente: Reportes de abril de 2004 de *Security Space (E-Soft Inc)*, *http://www.securityspace.com/*

5.7 Consideraciones acerca de IPv6

En IPv6, los servicios de cifrado, autenticación e integridad de los datos a nivel de red son proveídos por IPsec, empleando para ello criptografía simétrica. El intercambio de claves secretas se realiza en base al algoritmo *Diffie-Hellman*, con algunas adiciones para contrarrestar ataques tipo *man-in-the-middle* [30].

APPKEY no solo representa una alternativa para la determinación y distribución de claves secretas en IPv6, si no que además su uso implica que la provisión de servicios de seguridad puede ahora situarse por completo en la propia infraestructura de red. Una aplicación simplemente tendría que obtener del DNS la dirección IP y la clave pública del destinatario, para luego iniciar transmisiones seguras empleando IPv6, el cual puede hacer uso de la clave pública APPKEY para negociar e intercambiar las claves simétricas.

5.8 Acerca del registro APPKEY

La versión original de DNSSEC [9] establecía que, además del uso convencional del RR KEY como clave asociada a la firma de una zona, el mismo podía ser empleado por otras aplicaciones, de forma equivalente a la presente propuesta APPKEY. Sin embargo, en [25] se señalan acertadamente las inconveniencias de usar el mismo RR para estos disímiles propósitos, y se prohíbe usar el RR KEY para publicar en el DNS claves de otras aplicaciones. En los actuales *drafts* de la nueva versión del protocolo, se reemplaza el RR KEY por DNSKEY, quedando restringido su uso exclusivamente para la firma de zonas [2-4].

Actualmente, APPKEY es todavía un concepto. Queda por describirse sus características particulares, por ejemplo el formato y las especificaciones de interacción con el protocolo DNSSEC. En este documento han sido omitidos estos y otros detalles a fin de enfatizar los principios de arquitectura de la propuesta.

6 Conclusiones

Las actuales implementaciones de PKI basadas en X.509/PKIX y en el tradicional enfoque de CAs, presentan aún estimables desafíos. Algunos de estos desafíos se deben a que X.509 ha sido extendido para cubrir muchos requerimientos que no formaban parte de su diseño y de su campo de acción original. En este proceso, las especificaciones han crecido en tamaño y complejidad, motivando a su vez nuevas exigencias. Esto ocurre sobre todo cuando se pretende reflejar, en el entorno inmaterial e impersonal de las comunicaciones electrónicas, las relaciones de confianza inherentemente complejas del mundo real.

Sin embargo, esta complejidad no necesariamente debe ser reducida a una serie de formatos y estándares igualmente sofisticados. La búsqueda de la PKI perfecta no debería imponer límites al desarrollo de una buena PKI. Existen muchos usuarios y aplicaciones que precisan de un menor grado de complejidad a cambio de mayor versatilidad. Un importante criterio de diseño de arquitectura aplicado a las redes define que cuando la complejidad no puede ser eliminada, esta debe ser confinada a aquellas partes de la red que sean capaces de soportarla [16].

En este sentido, el presente trabajo propone un modelo alternativo de PKI que, valiéndose del uso de los servicios de infraestructura de Internet, reduce la complejidad inherente al esquema de relaciones de confianza en terceras partes. Se define para ello un nuevo registro DNS, denominado APPKEY, que permite a las organizaciones publicar las claves asociadas a sus entidades. Las aplicaciones obtienen a través de DNSSEC las claves públicas legítimas de cada *host,* con la misma facilidad con la que obtienen sus direcciones IP, para luego establecer los diferentes servicios de seguridad. Se elimina la necesidad de certificados de CAs externas debido a que las claves públicas son autenticadas por las propias organizaciones autoritativas sobre el espacio de nombres de la infraestructura de red en la que se despliegan estos servicios de seguridad.

El modelo propuesto es relativamente sencillo y cubre los requerimientos fundamentales de un importante número de aplicaciones de seguridad. Brinda a las partes la posibilidad de localizar y recuperar las claves públicas de las entidades, confiere autenticación a las mismas y otorga a las organizaciones el control de los criterios de seguridad a ser aplicados, sin depender de políticas internas de las CAs.

Una PKI que pretenda un emplazamiento extendido en toda la red Internet debe cumplir además con otros requerimientos (servicio de directorio simple y eficaz, escalabilidad, naturaleza distribuida, alta disponibilidad y balance de carga). El DNS tiene todas estas características y el modelo propuesto las aprovecha al máximo, sin agregar una sobrecarga significativa al sistema bajo las actuales condiciones.

Se han señalado además algunas ventajas respecto a aplicaciones como SSL, desde el punto de vista de la latencia en la comunicación, medida en *round-trips*, y del tamaño de los mensajes. El aprovechamiento de *caching* del DNS, es además un importante factor que posibilita un mejor aprovechamiento del ancho de banda, así como un mejor desempeño computacional a nivel colectivo.

Referencias

1. Albitz, P. y Liu, C. *DNS and Bind - 4th Edition.* O'Really, (2001).
2. Arends, R., Austein, R., Larson, M., Massey, D. y Rose, S. *DNS Security Introduction and Requirements.* IETF Draft, draft-ietf-dnsext-dnssec-intro-09, (Febrero 2004).
3. Arends, R., Larson, M., Austein, R., Massey, D. y Rose, S. *Protocol Modifications for the DNS Security Extensions.* IETF Draft, draft-ietf-dnsext-dnssec-protocol-05.txt, (Febrero 2004).
4. Arends, R., Austein, R., Larson, M., Massey, D. y Rose, S. *Resource Records for the DNS Security Extensions.* IETF Draft, draft-ietf-dnsext-dnssec-records-07.txt, (Febrero 2004).
5. Ateniese, G. y Mangard, S. *A New Approach to DNS Security (DNSSEC).* Proceedings of the 8th ACM Conference on Computer and Communications Security, (Noviembre 2001).
6. Atkins, D. y Austein, R. *Threat Analysis Of The Domain Name System.* IETF Draft, draft-ietf-dnsext-dns-threats-07.txt, (Abril 2004).
7. Boeyen, S. *Internet X.509 Public Key Infrastructure Repository Locator Service.* IETF Draft, draft-ietf-pkix-pkixrep-02.txt, (Septiembre 2003).
8. Chadwick, D. *Deficiencies in LDAP when used to support PKI.* Communications of the ACM, Vol. 46, No. 3, (Marzo 2003).
9. Eastlake, D. *Domain Name System Security Extensions.* IETF RFC-2535, (1999).
10. Ellison, C. y Schneier, B. *Ten Risks of PKI.* Computer Security Journal, Vol. XVI, No. 1, (2000).
11. Ellison, C. *Naming and certificates.* Proceedings of the tenth conference on Computers, freedom and privacy, ACM Press, (2000).
12. Freier, A., Karlton, P. y Kocher, P. *The SSL Protocol - Version 3.0.* Netscape Communications, (1996)
13. Gerck, E. *Overview of Certification Systems.* The Bell, Vol. 1, No. 3, (Julio 2000).
14. Gudmundsson, O. *Delegation Signer Resource Record.* IETF Draft, draft-ietf-dnsext-delegation-signer-15.txt, (Junio 2003).
15. Gudmundsson, O. *DNSSEC and IPv6 A6 aware server/resolver message size requirements.* IETF RFC-3226, (Diciembre 2001).
16. Hallam-Baker, P. *Trust Assertion XML Infrastructure.* Proceedings of the 1st Annual PKI Research Workshop, (Abril 2002).
17. Housley, R., Polk, W., Ford, W. y Solo, D. *Internet X.509 Public Key Infrastructure Certificate and CRL Profile.* IETF RFC-3280, (Abril 2002).
18. ITU-T. *Recommendation X.500: Information Technology - Open Systems Interconnection - The Directory: Overview of Concepts, Models and Service.* ITU-T, (1993).
19. ITU-T. *Recommendation X.501: Information Technology Open - Systems Interconnection - The Directory: Models.* ITU-T, (1993).
20. ITU-T. *Recommendation X.509: Information Technology - Open Systems Interconnection - The Directory: Autentication Framework.* ITU-T, (1988).
21. ITU-T. *Recommendation X.509 (1997 E): Information Technology - Open Systems Interconnection - The Directory: Autentication Framework.* ITU-T, (Junio 1997).
22. Josefsson, S. *Network Application Security Using The Domain Name System.* Royal Institute of Technology - Stockholm, Sweden, (2001)
23. Jung, J., Sit, E., Balakrishnan, H. y Morris, R. *DNS Performance and the Effectiveness of Caching.* IEEE/ACM Transactions on Networking, (Octubre 2002).
24. Kent, S. *How Many Certification Authorities Are Enough?.* Proceedings of MILCOM (unclassified papers), (Noviembre 1997).
25. Massey, D. y Rose, S. *Limiting the Scope of the KEY Resource Record (RR).* IETF RFC-3445,(2002).
26. Myers, M., Ankney, R., Malpani, A., Galperin, S. y Adams C. *Online Certificate Status Protocol - OCSP.* IETF RFC-2560, (Junio 1999).
27. Mockapetris, P. *Domain Names - Concepts and facilities.* IETF RFC-1034, (1987).
28. Mockapetris, P. *Domain Names - Implementation and Specification.* IETF RFC-1035, (1987).
29. Renfro, S. *VeriSign CZAG: Privacy Leak in X.509 Certificates.* Proceedings of the 11th USENIX Security Symposium, (Agosto 2002).
30. Stallings, W. *Cryptography and Network Security.* Prentice Hall, (1998).
31. Tanenbaum, A. *Computer Networks.* Prentice Hall, (2002).
32. Wahl, M., Howes, T. y Kille, S. *Lightweight Directory Access Protocol (v3).* IETF RFC-2251, (1997).

Mecanismos de conhecimento zero empregados por esquemas de chave pública

Vinicius G. Ribeiro
Coordenação de Pesquisa – Centro Universitário La Salle UNILASALLE(UFRGS)
Av. Victor Barreto, 2288 – 91.501-970 – Canoas – RS – Brasil
Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 92.010-000 – Porto Alegre – RS – Brasil
vribeiro@inf.ufrgs.br

Rafael Campello
Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 92.010-000 – Porto Alegre – RS – Brasil
campello@inf.ufrgs.br

e

Raul F. Weber
Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 92.010-000 – Porto Alegre – RS – Brasil
weber@inf.ufrgs.br

Abstract

This paper presents a comparative study among zero-knowledge mechanisms of public-key cryptography schemes. Special emphasis is given to a new proposed scheme, which computational problem is not based in Number Theory, but in problems of Differential Equations – that allows a simple authentication mechanism.

Keywords: Computer Security, Public-key Cryptography, Zero-knowledge.

Resumo

Este artigo apresenta um estudo comparativo dos esquemas de conhecimento zero empregados em alguns esquemas de criptografia de chave pública. Especial foco é dado a um novo esquema proposto, cujo problema computacional não é baseado na Teoria dos Números, mas em problemas das Equações Diferenciais – o que permite grande simplificação em seu mecanismo de autenticação.

Palavras-chave: Segurança Computacional, Criptografia de Chave Pública, Conhecimento Zero.

1. INTRODUÇÃO

Esquemas de chave pública têm chamado a atenção da comunidade acadêmica desde a sua criação – sendo o primeiro esquema Diffie-Hellman, em 1976. Tais esquemas têm, como base matemática, problemas de difícil resolução [8].

O objetivo de um desenvolvedor de sistemas é dotar, no sistema de informações em desenvolvimento, serviços de segurança. E dentre os diversos serviços de segurança, a autenticação ocupa um espaço relevante. Em se pensando na virtualização dos sistemas, nem sempre se pode contar com uma garantia de que, na outra extremidade, a outra parte seja quem afirma ser – e, por conseqüência, ter acesso a diversos serviços do sistema. Na verdade, isso se constitui em um dos grandes problemas existentes na criptografia: em uma comunicação, como garantir autenticação de uma das partes? Por autenticação de entidade, entende-se o serviço que permite que cada parte envolvida em uma comunicação possa assegurar-se que é quem afirma ser[10].

Considerando-se que, no campo dos esquemas de chave pública, há diversas formas de uma das partes se identificar, o presente artigo compara o mecanismos de autenticação de um tipo específico empregados por esquemas de chave pública. O tipo de mecanismo específico de interesse é a prova de conhecimento zero.

O presente trabalho está organizado da seguinte forma: na seção dois, é apresentado o foco do trabalho; na seção 3, são descritos os principais esquemas de conhecimento zero, sendo apresentado de forma mais detalhada o esquema Rafaella, proposto recentemente. A quarta seção apresenta o estudo comparativo e na última seção são apresentadas as considerações finais e trabalhos futuros.

2. CONSIDERAÇÕES SOBRE AUTENTICAÇÃO

A literatura emprega extensa terminologia no que se refere a mecanismos de autenticação - ou seja, o mecanismo de segurança que garante que cada parte é quem afirma ser [10, 11]. Na maior parte dos casos, considera método de autenticação como método para identificação de uma ou mais partes envolvidas; em outras, refere-se a provas de conhecimento zero. Tipicamente, os mecanismos empregados envolvem operação de verificação, nas qual se constata – dependendo do resultado obtido por essa operação – se a entidade é ou não quem alega ser. Algumas das características dos métodos empregados referem-se a situações onde a informação pode tornar-se ilegível, caso a parte não seja quem afirma ser.

O quadro a seguir apresenta os esquemas de Chave Pública, o problema que gera a dificuldade de descobrir a Chave Privada - aqui identificado como "Elemento de Dificuldade", bem como as funcionalidades que o esquema fornece.

Esquema de Chave Pública	Elemento de dificuldade					Funcionalidade				
	A	B	C	D	E	1	2	3	4	5
DSS		*						*	*	
ESIGN				*				*	*	
Feige-Fiat-Shamir	*				*			*		*
Guillou-Quisquater	*									*
Schnorr		*						*		*
Rafaella			*			*	*			*

Quadro 1 - Quadro comparativo dos principais esquemas de chave pública com alguma forma de identificação
Fonte: Elaborado pelos autores, com base no trabalho efetuado.

onde:

A - Fatoração de Números Inteiros
B - Logaritmo Discreto
C - Polinômio Quadrático/Cúbico
D - Resíduo Quadrático Módulo n
E - Determinação da função original

1 - Cifragem de Mensagens
2 - Decifragem de Mensagens
3 - Assinatura Digital
4 - Verificação de Mensagens
5 - Identificação/Autenticação

De modo geral, os algoritmos de esquemas de Chave Pública são baseados em problemas de grande dificuldade de fatoração de números, ou ainda para a determinação de expoentes de operações modulares, ou ainda a determinação de escalares que multiplicaram determinado ponto. Assim, assume-se que a fatoração, a determinação de escalares em curvas elípticas e o cálculo do logaritmo discreto são computacionalmente inviáveis, visto a não existência - ou conhecimento - de algoritmo que o resolva em tempo polinomial.

Por exemplo, acrônimo para *Digital Signature Standard*, o DSS é empregado apenas para fins de assinatura digital, foi proposto pelo Instituto Nacional de Padrões e Tecnologia (*National Institute of Standards*

and Technology - NIST), em dezembro de 1994. Baseia-se na dificuldade de calcular o Logaritmo Discreto de determinado número módulo um primo. A demonstração encontra-se completa em Terada [12].

O ESIGN é um esquema de assinatura digital, criado pela equipe da NTT, liderado por T. Okamoto. Segundo os autores, é tão seguro quanto o RSA, mas seria mais rápido do que esse - dadas as condições similares de tamanhos das chaves e assinaturas. Já foram propostas variações para esse esquema, com o intuito de torná-lo de mais difícil quebra. Segundo os autores, esse algoritmo pode ser estendido para trabalhar com curvas elípticas. Mais informações podem ser obtidas em Schneier [11].

Schnorr é um esquema de assinatura digital e autenticação, o qual emprega idéias baseadas em ElGamal, Fiat-Shamir, e o protocolo interativo de David Shaum, Jan-Hendrick Evertse e Jeroen van de Graaf. Tem sua segurança baseada na dificuldade de calcular logaritmos discretos. Encontra-se a demonstração desse esquema em Terada [12].

Já o esquema Rafaella trabalha com funções. Assim, os procedimentos para determinação de mensagem original constituem-se nos problemas da área das Equações Diferenciais. Dada uma função original, e aplicando-se determinadas transformações, é um trabalho muito difícil efetuar a transformação inversa, ou mesmo determinar qual a função original, mesmo empregando-se técnicas avançadas - como o emprego de operadores infinitesimais de 1ª ordem de Lie, por exemplo.

Para o presente trabalho, são considerados os mecanismos de conhecimento zero, empregados pelos esquemas de chave pública. Por "conhecimento zero", entende-se a forma pela qual um participante legítimo de um protocolo pode ser convencido da identidade do outro participante - normalmente, por intermédio da solução de algum problema da Teoria dos Números, ou da Análise Combinatória [4].

A seção seguinte apresenta os principais esquemas de conhecimento zero.

3. AUTENTICAÇÃO POR PROVA DE CONHECIMENTO ZERO

Há diversas formas de classificar os esquemas de chave pública que empregam algum tipo de autenticação. No presente trabalho, interessa o estudo e a classificação dos métodos empregados em esquemas que ofereçam a funcionalidade de identificação, buscando identificar a forma pela qual é possível efetuar a identificação é realizada – mesmo sem alguma informação do outro usuário -, e o que ocorre caso um usuário não autorizado resolva passar-se por um participante do esquema. Trata-se, assim, de identificar os problemas matemáticos envolvidos nesses serviços, e como eles são empregados.

Um dos mais comuns problemas em criptografia diz respeito a um participante autorizado obter provas que o outro participante realmente é quem afirma [6, 11]. Implicitamente, essa colocação envolve a noção de comunicação entre partes, bem como as regras para essa comunicação – ou seja, um protocolo. Tipicamente, os esquemas de chave pública que empregam serviços de autenticação e semelhantes têm regras bem definidas para, em algum passo do protocolo, possibilitar a verificação da prova que identifica a outra parte. Pelo menos, duas partes se envolvem: o proponente da prova e o verificador.

Há diversas formas de realizar comprovações da identidade de um participante. Há protocolos criptográficos que identificam pessoas, através de testemunhos; há esquemas onde são geradas e verificadas assinaturas digitais; há também os esquemas onde não se conhecem detalhes do outro participante, mas há formas de mostrar que esse participante é quem afirma ser, por apresentar alguma prova: são os chamados esquemas de conhecimento zero [1].

O primeiro esquema baseado nesse últimos foi formalmente proposto por Goldwasser, Micali e Rackoff, em 1985 [2], considerando interação entre as partes. Esquemas de conhecimento zero podem classificados em quatro grandes grupos, segundo o modelo formal de computação proposto pelos autores citados anteriormente [5]:

- conhecimento zero perfeito;
- conhecimento zero com verificador honesto;
- conhecimento zero computacional; e
- conhecimento zero estatístico.

Diferente de empregar apenas uma função unidirecional para revelar determinada informação, tais esquemas provêem a funcionalidade de permitir que o verificador venha a conhecer algo que não poderia de ninguém, exceto pelo proponente da prova. O emprego de uma função unidirecional apenas possibilita informar que o proponente tem uma parte da informação, sem fornecer modo algum de determinar qual é a informação [11].

O esquema de prova de conhecimento zero perfeito tem, como exemplo mais fiel, o esquema proposto por Schnorr; já o exemplo que se baseia na prova de conhecimento zero com verificador honesto tem como exemplo o esquema heurístico Fiat-Shamir [11].

A seguir, são apontadas as descrições do funcionamento dos esquemas – ou seja, por que razão uma das partes pode vir a identificar a outra, mesmo sem estar em sua presença física.

3.1 Esquemas baseados em problemas da Teoria dos Números

Os esquemas de criptografia de chave pública pertencentes a essa família são baseados, em geral, nos problemas de fatoração de grandes números e logaritmo discreto. Com relação aos esquemas que empregam para fins de autenticação, tem-se como exemplos representativos os esquemas Guillou-Quisquater e Feige-Fiat-Shamir. No caso dos esquemas de criptografia de chave pública baseados em resolução de equações diferenciais, tem-se como exemplo representativo o esquema Rafaella, descrito em seção posterior.

O esquema proposto inicialmente por Amos Fiat e Adi Shamir é basicamente um esquema de autenticação e de assinaturas digitais. Posteriormente, com a participação de Uriel Feige, esse esquema foi alterado para constituir uma chamada prova de identidade com conhecimento zero. Atualmente, é considerado o melhor esquema para prova de identidade com conhecimento zero [11, 13]. Para esse esquema, é necessário empregar um árbitro - cuja principal função é realizar a escolha de um número randômico módulo n (produto de dois números primos grandes). O número n pode ser compartilhado com todos os envolvidos - interessados em se autenticar [11].

Para gerar as Chaves Pública e Privada de Alice, o árbitro de confiança escolhe um número v - v é um resíduo quadrático módulo n. Em outras palavras, escolhe v, tal que

$$x^2 = v \bmod n$$

tenha uma solução, e v^{-1} mod n exista. Essa última expressão é a chave pública. Calcula-se, então, s - o menor valor encontrado nas raízes da expressão

$$s = (\sqrt{(1/v)}) \bmod n,$$

que é a chave privada.

Para fins de identificação, tem-se o seguinte protocolo:

1 - Alice escolhe um número randômico r, sendo r menor do que n. Ela calcula $x = r^2$ mod n, e manda x para Bob.

2 - Bob manda um bit randômico b para Alice.

3 - Dependendo do valor do bit, Alice manda um valor diferente a Bob. Se o valor do bit for 0, Alice manda r para Bob. Se o valor do bit for 1, Alice manda y - sendo y = r * s mod n.

4 - Se o valor de b for igual a 0, então Bob verifica que $x = r^2$ mod n - o que prova que Alice conhece o valor de $\sqrt{x}$. Se o valor do bit for igual a 1, então Bob verifica que $x = y^2 * v$ mod n, o que prova que Alice conhece o valor de $\sqrt{(x/v)}$.

Destaca-se que esse protocolo deve ser realizado diversas vezes, para que Bob fique convencido que Alice conhece o valor de s - o qual depende do valor de v, escolhido pelo árbitro de confiança, que conhece a ambos. Em outras palavras, se ela desejar enganar Bob, terá apenas 50 % de chances de fazê-lo, em 1 tentativa. Em t tentativas, as chances dela são de 1 em 2^t. Ou seja, esse protocolo busca identificar a primeira pessoa(Alice) perante uma segunda (Bob).

Posteriormente, esse esquema foi alterado para um esquema de realmente executar a identificação: basicamente, Bob não envia apenas um bit, mas uma string de bits ($b_1b_2b_3..b_k$), e Alice realiza o cálculo $y = r * (s_1b_1 * s_2b_2 *... * s_nb_k)$ mod n - ou seja, s só é multiplicado, se o bit correspondente tiver calor 1; se tiver valor 0, não o será. De modo análogo, Bob verifica que $x = y * (v_1b_1 * v_2b_2 *... * v_nb_k)$. Da mesma forma, deve ser repetido t vezes, sendo a chance de que Alice engane Bob é de uma em 2^{kt}.

Modificações posteriores permitiram que esse esquema se tornasse um esquema de assinatura digital, e ainda, esquemas envolvendo mais de duas pessoas. Há autores que colocam a possibilidade de se embutir informação no protocolo, o que possibilitaria a operação de cifragem [11].

O esquema Guillou-Quisquater

O esquema Guillou-Quisqueter é uma outra forma de realizar um método de identificação, como conhecimento zero, mas com um menor número de iterações entre os participantes do esquema[13]. Contudo pode, com algumas alterações, ser convertido em um esquema de assinatura verificável publicamente. Esse esquema é uma extensão do Esquema Fiat-Shamir, mas com menor troca de mensagens, e se baseia na dificuldade da Fatoração de Números Primos.

Alice remete suas credenciais - J - para Bob. Essas credenciais podem ser uma grande *string* contendo dados diversos, como se fora um cartão de crédito , ou *smart card* - tais como nome, data de nascimento, nome do cartão, data de validade, número da conta bancária, entre outros. Qualquer outra informação que possa se tornar pública, é um expoente v e um módulo n - produto de dois grandes números primos. Já a chave privada é B, calculada tal que

$$JC_P^v \equiv 1 \pmod n.$$

Tendo remetido as suas credenciais para Bob, agora Alice deseja provar que essas credenciais lhe pertencem. Para tanto, deve provar que conhece "C_P", a Chave Privada, sem expô-la.

O esquema empregado para identificação é executado em três fases, e pode ser resumido da seguinte forma:

A - Fase de determinação dos parâmetros gerais

1 - Um terceiro confiável escolhe dois números primos grandes, e calcula n = p*q.

2 - A entidade confiável escolhe v - público - , tal que $v \leq 3$, e que seja relativamente primo a (p-1)*(q-1).

3 - A entidade confiável calcula s - secreto -, tal que $s = v\text{-}1 \bmod (p-1)*(q-1)$, e disponibiliza publicamente os parâmetros "v" e "n".

B - Fase de escolha dos parâmetros individuais

1 - Cada usuário recebe uma identificação única de usuário, I_u. Calcula-se uma identificação redundante J_u, a partir de uma função pública e conveniente, tal que $J_u = F(I_u)$, que J_u seja relativamente primo a "n", e que $1 \leq J_u \leq n$. "J_u" é a credencial de cada usuário.

2 - A entidade confiável calcula o segredo "s" para cada usuário. O objetivo será a posterior comprovação da identidade pelo conhecimento do segredo, associado à Chave Pública.

C - Fase de Identificação

1 - Alice escolhe um número randômico "r", tal que $1 < r < n-1$, e calcula o chamado testemunho $T \equiv r^v \pmod n$. Remete, então, T a Bob, juntamente com sua identificação única - fornecida por um terceiro confiável.

2 - Bob remete a Alice um desafio "e", tal que $1 \leq e \leq v$.

3 - Alice realiza a verificação para constatar se "e" se encontra nos limites. Caso se encontre no limite, ela envia a Bob a resposta

$$y = r\,(s_A)^e \bmod n.$$

4 - De posse da função f - que é pública -, Bob calcula $J_A = f(I_A)$, e $z = J^e_A\, y^v \bmod n$. Caso z seja diferente de zero, e for igual a x, Bob aceita a identidade de Alice.

Cabe questionar por que Bob aceita a identidade de Alice, caso z = x. Observa-se que

$$\begin{aligned} z &= J^e_A\, y^v \bmod n \\ &= J^e_A\, (r(s_A)^e)^v \bmod n \\ &= r^v (J_A\, (s_A)^v)^e \bmod n \\ &= r^v ((s_A)^s (s_A)^v)^e \bmod n \\ &= r^v (1)^e \bmod n \\ &= r^v \bmod n = x. \end{aligned}$$

Destaca-se a dificuldade de determinar x^{-v}, sem fatorar n - pois é computacionalmente inviável, sendo equivalente ao problema de se decifrar RSA sem conhecer a chave secreta. A condição de z ser diferente de zero decorre da possibilidade que um adversário, passando-se por Alice, escolha r = 0 - o que incorreria em z = 0 (Terada, 2000). Esse esquema também pode ser alterado para possibilitar assinatura digital [11].

Na próxima seção, é apresentado um esquema baseado em paradigma não da Teoria dos Números, mas da área das Equações Diferenciais.

3.2 Esquemas baseados em grupos de Lie

Diferente dos esquemas tradicionais, baseados fortemente em grupos de Galois, essa família é baseada nos chamadas simetrias de Lie. O primeiro esquema dessa família foi proposto por Ribeiro e Weber, em 2004 [9], e é baseado na dificuldade de se identificar translações de variáveis sobre funções, considerando as propriedade das simetrias de Lie. A chave pública, nesse esquema, é uma função construída a partir da chave privada do usuário; a chave privada é um número complexo com as partes real e imaginária diferentes de zero. O problema inverso é a dificuldade é a determinação qual seria a possível equação diferencial que originaria as funções que transitam em aberto. Ao atacante, cabe apenas duas alternativas: determinar essa equação diferencial ou a varredura de todo o plano complexo. Empregado para realizar a cifragem de mensagens, inclui a funcionalidade de prova interativa de conhecimento zero. Contudo, todas as operações constituem procedimentos sobre funções, e não sobre números - razão pela qual o esquema Rafaella emprega programas de processamento simbólico (*Derive*, *SimbMath*, *Mathematica* ou *Maple V*®).

No esquema proposto, o processo de autenticação é realizado através do cálculo de dois argumentos auxiliares; o primeiro é uma função contínua, e o segundo, um número complexo – podendo até ser empregado um operador infinitesimal dos grupos de Lie. O primeiro argumento, denominado de *argumento de autenticação*, consiste em uma função contínua, formado por combinações lineares entre potências das chaves públicas de ambos os participantes. Sobre essa função, o proponente da prova deve aplicar sua chave privada, a fim de produzir alterações sobre uma nova função. A nova função obtida é, então, utilizada por parte do verificador , para verificar a autenticidade do proponente da prova, através de um teste de autenticidade. A construção e a verificação são apresentadas a seguir:

Basicamente, é construída uma função em duas variáveis, empregando-se as chaves pública de cada participante, e a chave privada de participante verificador. O cálculo do chamado *argumento de autenticação* consiste em uma função contínua, formado pelo produto das chaves públicas de ambos os participantes, somados com uma função cuja raiz é a sua própria chave privada. Por exemplo,

$$aa = C_{pub}A * C_{pub}B + (y - C_{priv}A)^n.$$

Sobre essa função, o receptor deve aplicar sua chave privada, a fim de reduzir a uma expressão cuja validação final será efetuada pelo verificador. Assim, o receptor - Bob - retira o fator das chaves públicas, ao aplicar a sua chave privada

$$ta = aa(C_{priv}B) = C_{pub}A * 0 + (y - C_{priv}A)^n = (y - C_{priv}A)^n$$

, resultará a expressão ta - que nada mais é senão uma função cuja raiz é a chave privada de Alice.

A nova função obtida é, então, utilizada por parte do emissor, para verificar a autenticidade do receptor, através de um simples teste de autenticidade: ao aplicar na função restante a chave privada, deverá resultar no valor zero. Essa prova reconhece que o receptor realmente é quem afirma ser, por ter aplicado no argumento de autenticação a correta chave privada. Ou seja,

$$ta = 0.$$

Caso o número resulte nulo, a autenticidade do receptor é constatada; caso contrário, o verificador interrompe o processo, não permitindo que a mensagem final seja conhecida. Dessa forma, apenas o receptor autorizado poderá, ao final do processo, recuperar a mensagem original.

Esse processo é ilustrado no quadro a seguir, onde os passos referentes à autenticação da mensagem são destacados com um "S".

Procedimento do passo	Processo de Autenticação (S/N)
1. conversão da mensagem	N
2. escolha da forma da função e produção de argumentos de autenticação	S
3. aplicação do 1º deslocamento no plano complexo	N
4. envio da mensagem cifrada	N
5. aplicação do 2º deslocamento	S
6. envio da mensagem obtida	N
7. aplicação do 1º deslocamento inverso e verificação da autenticidade do proponente da prova	S
8. envio da mensagem obtida	N
9. aplicação da 2ª mudança inversa e verificação da autenticidade do verificador	S
10. recuperação dos caracteres originais	N

Quadro 2: Passos do esquema Rafaella onde se empregam procedimentos de autenticação

Na próxima seção, são levantadas considerações gerias sobre os esquemas.

4. CONSIDERAÇÕES FINAIS SOBRE OS ESQUEMAS DE CONHECIMENTO ZERO

Dentre diversos esquemas de chave pública que empregam prova de conhecimento zero, com o intuito de autenticar a outra parte, foi construído o quadro a seguir.

Esquema de Chave Pública	**Operações**	**Elementos sobre o qual são realizadas transformações**	**Condição de Aceitação**	**Área matemática do(a)**	
				problema inverso	**condição de aceitação**
Guillou-Quisquater	Algorítmicas	Números	Bob aceita a identidade de Alice, se z<> 0, e z = x	Teoria dos Números	Teoria dos Números
Feige-Fiat-Shamir	Algorítmicas	Números	Dependendo do valor do bit, Alice manda um valor diferente a Bob. Se 0, Alice manda r. Se 1, Alice manda y. Se b = 0 então Bob verifica que $x = r^2 \bmod n$ (Alice conhece $\sqrt{x}$). Se b = 1 então Bob verifica que $x = y^2 * v \bmod n$ (Alice conhece $\sqrt{(x/v)}$).	Teoria dos Números	Teoria dos Números

Rafaella	Algébricas	Funções	Bob retorna a Alice uma função recebida de Alice, e devidamente operada por Bob. Se Alice obtiver o valor nulo ao inserir sua chave privada, aceita a identidade de Bob	Equações Diferenciais	Teoria dos Números

Quadro 3: Comparação entre esquemas de conhecimento zero dos esquemas de chave pública
Fonte: elaborado pelos autores, com base no estudo realizado.

Como é possível observar do quadro acima, os esquemas Feige-Fiat-Shamir, Guillou-Quisquater e Rafaella empregam problemas de determinação ou comparação com valores numéricos - sendo que o esquema Rafaella difere dos demais por seu mecanismo operar sobre funções, e seu problema inverso pertencer à área das equações diferenciais. Aparentemente, mecanismos de prova de conhecimento zero dificilmente diferirão de realizar operações - sejam numéricas, sejam algébricas - e posterior comparação com valores. Uma diferença do esquema Rafaella é o emprego de programas de processamento algébrico ou simbólico.

Considerando as informações de Ribeiro [8], o seu problema inverso é de difícil solução [2, 7]. Em Ribeiro, são levantadas considerações sobre o desempenho desse novo esquema - graças à dependência de empregar os programas supracitados. Outra diferença desse esquema sobre os outros é funcionalidade: sua principal funcionalidade é a cifragem e decifragem de mensagens, e não especificamente a autenticação dos participantes - embora, para essa funcionalidade, seja empregado de modo simétrico: tanto Alice identifica Bob, quanto Bob identifica Alice.

5. REFERÊNCIAS BIBLIOGRÁFICAS

[1] Blum, Manuel; De Sanctis, Alfredo; Micali, Silvio & Persiano, Giuseppe, *Non-interactive zero knowledge*, march 1995.

[2] Blumen, G. W. and Kumei, S. *Symmetries and differential equations*. New York: Springer-Verlag. 1989.

[3] Goldreich, Oded & Kahan, Ariel. "How to construct zero knowledge proofs systems for NP", Journal of Cryptology, march 1999.

[4] Koblitz, Neal. *Algebraic Aspects of Cryptography*. Berlin, Springer-Verlag. 1999.

[5] Mao, W. *Modern Cryptography – Theory and Practice*. Upper Saddle River: Prentice Hall. 2004.

[6] Menezes, A.; van Oorschot, P. & Vanstone, S. *Handbook of Applied Cryptography*. Boca Raton: CRC Press. 1996.

[7] Olver, P. *Applications of Lie Groups to Differential Equations*. New York: Springer. 2000.

[8] Ribeiro, V.G. *Rafaella: um esquema de criptografia de chave pública baseado em um novo paradigma matemático*. Tese de doutorado. Porto Alegre: Programa de Pós-Graduação em Computação, Universidade Federal do Rio Grande do Sul, Porto Alegre. 2004.

[9] Ribeiro, Vinicius Gadis & Weber, Raul Fernando. Problemas matemáticos para esquemas de criptografia de chave pública. *Anais do SBRC/Wseg 2004*. Porto Alegre: Instituto de Informática/UFRGS. 2004. p. 101-112.

[10] Stallings, W. *Cryptography and Network Security: Principles and Practice*. Upper Saddle River: Prentice Hall. 1999.

[11] Schneier, Bruce. *Applied Cryptography - Protocols, Algorithms and Source Code in C*. New York: John Wiley & Sons. 1994.

[12] Terada, Routo. *Segurança de Dados - Criptografia em redes de computador*. São Paulo: Ed. Blücher. 2000.

[13] Weber, Raul F. Criptografia Contemporânea. In: Simpósio de Computadores Tolerantes a Falhas, 6, 1995, Canela. *Anais*... Porto Alegre: Instituto de Informática da UFRGS, 1995. pp. 7-32.

Arquitetura Multiagente Improvisacional: Transformando Planejamento em Improvisação e Introduzindo Improvisação nos Processos de Solução de Problemas

Márcia Cristina Moraes
Universidade Federal do Rio Grande do Sul, Programa de Pós-Graduação em Computação
Pontifícia Universidade Católica do Rio Grande do Sul, Faculdade de Informática
Porto Alegre, Brazil, 90619-900
mmoraes@inf.ufrgs.br, mmoraes@inf.pucrs.br

e

Antônio Carlos da Rocha Costa
Universidade Federal do Rio Grande do Sul, Programa de Pós-Graduação em Computação
Pontifícia Universidade Católica de Pelotas, Escola de Informática
Pelotas, Brazil, 96010-000
rocha@atlas.ucpel.tche.br

Abstract

This paper proposes the understanding of improvisational agents as rational agents, through the presentation of an improvisational multi-agent architecture that incorporates improvisation processes in both, tactical level related to the building of agent's course of action, and strategic level related to problem-solving. In the first case, improvisation brings alternatives course of action executions, considering agent's plans as intentions. In the second case, improvisation brings the capability to handle problems that weren't anticipated in the planning of its course of action. The paper opposes this approach to probabilistic reasoning and re-planning, usually used in such situations. Improvisation allows agents to give rapid answers to unexpected situations, independently of having explicit knowledge directly applicable to such situations. To do that, agents use the resources readily available to them, using a process based on analogy by similarity.

Keywords: Artificial Intelligence, Autonomous Agents and Multi-Agent Systems, Planning and Scheduling, Agent Architectures, Improvisational Agents

Resumo

Este artigo propõe o entendimento de agentes improvisacionais como agentes deliberativos, através da apresentação de uma arquitetura multiagente improvisacional que incorpora processos de improvisação tanto no nível tático da condução do curso de ação do agente quanto no nível estratégico da resolução de problemas. No primeiro caso, a improvisação visa proporcionar alternativas de execução a um determinado curso de ação, considerando os planos dos agentes como intenções. No segundo caso, ela traz a habilidade para tratar sub-problemas que não foram antecipados no planejamento do seu curso de ação. O artigo contrasta essa abordagem com as de raciocínio probabilístico e de re-planejamento normalmente utilizadas nessas situações. A improvisação possibilita aos agentes fornecerem respostas rápidas para situações inesperadas, independentemente de terem conhecimento explícito aplicável a tais situações. Os agentes usam os recursos diretamente disponíveis para eles, com base em um processo de analogia por similaridade.

Palavras-Chave: Inteligência Artificial, Agentes Autônomos e Sistemas Multiagentes, Planejamento, Arquiteturas de Agentes, Agentes Improvisacionais

1 Introdução

Tradicionalmente, um agente artificial tem as capacidades de raciocínio, aprendizagem e comunicação com outros agentes. O raciocínio é utilizado por agentes racionais para solucionar problemas de uma maneira correta e completa. Para fazer isto, os agentes racionais planejam seu curso de ação antecipadamente, tentando prever o futuro. Contudo, ambientes reais podem se modificar enquanto um agente está raciocinando sobre como alcançar algum objetivo, e estas mudanças podem invalidar as suposições sobre as quais o raciocínio do agente está fundamentado. Desta maneira, os agentes que operam em ambientes reais e dinâmicos precisam estar atentos a muitas situações inesperadas, que normalmente não surgem de maneira seqüencial, ou seja, os agentes precisam raciocinar sobre as suas ações [1]. Este raciocínio implica que os agentes devam saber quando novos fatos e oportunidades acontecem e como se adaptar à situação atual. Uma alternativa amplamente utilizada para dar conta desses ambientes dinâmicos é modelá-los através de recursos probabilísticos, ou então de replanejamento das ações, e realizar adaptação através de processos de aprendizagem [2]. Uma maneira alternativa, proposta por Hayes-Roth e Doyle [3], que é a adotada neste trabalho, é o uso de processos de improvisação. Em frente a situações inesperadas, que exigem respostas rápidas e espontâneas, as pessoas usam a sua capacidade de improvisação. Neste tipo de situação, muitas vezes é melhor improvisar do que re-planejar.

A capacidade de produzir respostas espontâneas é particularmente importante para agentes de interface que podem ser personificados através de personagens animados ou figuras humanas. Alguns pesquisadores têm mostrado que os usuários aplicam regras sociais aos computadores [4][5]. Deste modo, assim como os agentes humanos, os agentes racionais precisam apresentar comportamentos credíveis, interessantes e coerentes, trazendo a ilusão de visa para os agentes e fazendo o usuário suspender suas descrenças em relação aos agentes [6]. Para realizar isto, os agentes precisam agir de maneira credível tanto em frente a situações esperadas como em situações inesperadas.

Para serem credíveis no primeiro caso, os agentes podem usar o que chamamos de improvisação implícita. Por improvisação implícita, se entende a habilidade de incorporar cursos de ação previsíveis alternativos. Desta maneira, entendemos que planejamento se torna improvisação implícita, pois é mais razoável guiar agentes com um curso de comportamento abstrato do que dirigi-los através de um curso completo de ação. Improvisação implícita é tratada usando a noção de planos como intenção, ao invés de planos como programas, numa abordagem baseada em satisfação de restrições. No segundo caso, os agentes podem usar o que chamamos de improvisação explícita. Por improvisação explícita, se entende a habilidade de tratar um evento conhecido imprevisível. Com a improvisação explícita se tem o objetivo de introduzir o conceito de improvisação no processo de solução de problemas de agentes racionais, tratando improvisação, neste caso, como um processo baseado na analogia por similaridade.

Neste artigo apresentamos uma arquitetura multiagente improvisational que incorpora as mudanças de planejamento para improvisação implícita e a introdução de improvisação explícita na solução de problemas de agentes racionais, baseados em uma arquitetura BDI. A arquitetura a ser apresentada possui seus conceitos de improvisação fundamentados no teatro improvisacional. Trabalhos relacionados com improvisação e teatro improvisacional, tais como Teatro Virtual [7], Oz [8] e Improv [9], enfocam somente eventos conhecidos previsíveis, usando abordagens como planejamento e improvisação implícita. Todos estes trabalhos enfocaram o papel dos atores e não examinaram o papel do diretor. Nossa proposta é inovadora, pois aborda improvisação explícita baseada nas habilidades do diretor em tratar problemas que não foram antecipados no momento do planejamento. No artigo, enfocamos o papel do diretor na arquitetura multiagente improvisacional, considerando sua capacidade de solucionar problemas que não foram previstos no momento da construção do seu curso de ação.

2 Improvisação, Satisfação de Restrições e Analogia por Similaridade

De acordo com Spolin [10], improvisação está relacionada à espontaneidade de agir em um mundo em constante movimento. Desta maneira, a idéia de improvisação e de teatro improvisacional está implicitamente apegada à representação como um comportamento informal e espontâneo que é realizado sem preparação prévia. O teatro tradicional, por outro lado, entende que a representação sempre é algo pensado, algo organizado antecipadamente. Chacra [11] salienta que representação improvisada e planejada são pólos diferentes de uma mesma matéria, determinadas por graus que fazem a apresentação teatral mais ou menos formalizada ou improvisada. Se os atores têm a intenção de usar improvisação eles estão implicitamente integrados no que é

chamado de Teatro Improvisacional. Desta maneira, eles não preparam antecipadamente todas suas falas e gestos, eles consideram o momento de espontaneidade. O momento de espontaneidade age em dois momentos de improvisação: implícita e explícita. A improvisação implícita envolve improvisação de texto (que ocorre através de um conjunto de frases e comportamentos não verbais que são deixados em aberto na peça para livre improvisação dos atores) e improvisação de personalidade (que está relacionada à maneira como o ator interpreta um personagem, considerando suas características físicas e psicológicas). Entendemos que este tipo de improvisação é realizado através da satisfação das restrições relacionadas ao texto (curso de ação) do agente e da sua personalidade. A improvisação explícita envolve improvisação de solução de problemas que ocorre quando um agente não tem um plano que pode ser imediatamente aplicado a uma situação inesperada. Neste caso, o diretor é responsável por auxiliar os atores a encontrar solução para problemas não inesperados. O diretor precisa de respostas rápidas para situações inesperadas e usa os recursos disponíveis para produzir estas respostas. Uma abordagem que fornece respostas rápidas e usa experiência passada (recursos) para produzir novas soluções para problemas inesperados é a analogia por similaridade [12].

2.1 Improvisação Implícita como Satisfação de Restrições

A improvisação implícita trata da questão da mudança de abordagem de planejamento para improvisação. Para entendermos como o planejamento se torna improvisação temos que considerar que o mundo real não é estático e completamente conhecido. Desta maneira, os agentes devem levar em conta as mudanças que podem ocorrer no ambiente enquanto eles raciocinam sobre como alcançar seus objetivos, ou seja, eles devem considerar a situação corrente no momento de agir. Vários autores apresentam abordagens diferentes do planejamento tradicional para tratar esta questão [13] [14] [15] [16] [17]. Para estes autores, o planejamento clássico não pode ser aplicado a ambientes dinâmicos e complexos, e a abordagem mais adequada é a improvisação.

Do ponto de vista do planejamento, existem duas maneiras de ver um plano. No planejamento clássico, um plano é visto como um programa e em uma abordagem alternativa um plano é visto como uma intenção. De acordo com Pfleger e Hayes-Roth [18] na visão de planos como programas, um plano é um programa executável composto por ações primitivas que um agente pode executar a fim de agir. Logo, o planejamento é um tipo de programação automática, e a aplicação do plano consiste simplesmente na sua execução direta. Na outra visão, um plano é um comprometimento a um objetivo que guia, mas não determina as ações específicas que um agente executa. Desta maneira, o agente não pode executar diretamente o seu plano, mas somente pode executar comportamentos, cada um dos quais podendo estar mais ou menos consistentes com os planos.

A visão de planos como programas possuem várias limitações como possibilidade de complexidade exponencial e inadequação para um mundo caracterizado por eventos imprevisíveis [13]. Planos como intenções superam estas limitações, pois um plano é um recurso que guia, através de ações abstratas, o que o agente deve fazer.

Nós iremos utilizar a segunda abordagem de planos como intenções para mostrar como planejamento se torna improvisação. Para alcançar suas intenções as pessoas podem ter uma idéia do que elas devem fazer e isto está relacionado a uma série de limitações e oportunidades que são chamadas de restrições. O agente tem a liberdade de improvisar, considerando as restrições presentes no momento. Esta visão não indica somente os objetivos do agente, mas também algum conjunto de comportamentos possíveis para alcançar estes objetivos [18].

Na nossa visão um agente tem alguma intenção e esta intenção poderia ser descrita como um *script* que informa algumas ações do que deve ser feito e alguns comportamentos que satisfazem estas ações. Estas ações são abstratas, descrevendo procedimentos gerais para se alcançar uma intenção. O agente somente pode escolher o que fazer e como fazer quando está frente a alguma situação. Uma intenção poderia ser entendida como um objetivo que é representado através de um *script* de alto nível que é instanciado com ações concretas de acordo com a situação atual. Este *script* de alto nível e as ações concretas são representados como improvisações que são realizadas através de satisfação de restrições.

De acordo com Marriott [19], restrições são formalizações matemáticas de relacionamentos que podem ocorrer entre objetos, elas são representadas através de um domínio de restrições que especifica as constantes, as funções e os relacionamentos de restrições que podem ocorrer entre objetos. A arquitetura proposta possui duas classes de restrições: restrições de ordem (relacionadas à ordem na qual algum conteúdo deve ser organizado e apresentado) e restrições de comportamento (relacionadas ao processo de selecionar comportamentos apropriados para executar).

A representação da intenção é baseada no modelo de regras de produção. De acordo com Stefik [20], cada regra de produção possui duas partes, chamada parte-se e parte-então. A parte-se da regra consiste de condições a serem testadas. Se todas as condições da parte-se de uma regra são verdadeiras, as ações da parte-então são

executadas. A diferença fundamental entre a nossa representação e as regras de produção tradicionais é que as ações da parte-se são comportamentos abstratos, ou ações de alto nível, que irão ser transformadas em ações primitivas durante a improvisação. Escolhemos utilizar o modelo de regras de produção, porque consideramos que esta é a representação mais apropriada para intenções considerando a abordagem do teatro improvisacional.

2.2 Improvisação Explícita como Analogia por Similaridade

Na improvisação explícita, o agente deve fazer uso dos recursos disponíveis para solucionar um problema de forma correta e rápida. Uma abordagem que fornece um mecanismo para efetivamente conectar um raciocínio passado com uma experiência presente, é a analogia. Na analogia os aspectos conhecidos de um novo caso são comparados com aspectos correspondentes de casos antigos. O caso antigo que possui a melhor combinação de aspectos correspondentes pode ser assumido como a melhor fonte de evidência para estimar os aspectos não conhecidos do novo caso. Quanto maior à concordância entre as alternativas dos aspectos não conhecidos, mais forte a evidência para uma conclusão do novo caso. De acordo com Russell [12], a analogia por similaridade fornece um mecanismo baseado na teoria de determinações [21][22] capaz de produzir respostas plausíveis e rápidas para problemas existentes em ambientes complexos.

De acordo com Russell [12] [21], um modelo de analogia simplificado em um banco de dados é o seguinte: existe um alvo T descrito por *m* pares de atributo-valor, para o qual se quer encontrar o valor de outro atributo Q. Existem várias fontes S_1, ..., S_n (análogas) que têm valores para o atributo Q desejável bem como para os *m* atributos conhecidos para o alvo. A similaridade *s* é definida como o número de valores de atributos que combinam para um determinado alvo e fonte e a diferença é definida por $d = m\text{-}s$. Assumindo que existem *r* atributos relevantes para descobrir o valor de Q, $p(d, r)$ é definida como sendo a probabilidade que a fonte S, diferindo do seu alvo em *d* atributos, se combina com os *r* atributos relevantes. A suposição de que não existe informação relevante significa que todos os atributos são igualmente importantes para serem relevantes. Pode-se calcular $p(d,r)$ usando um argumento combinatório simples [12]. Seja N_m o número de escolhas de quais atributos são relevantes tais que S combina com T em tais atributos. Seja *N* o número total de escolhas de quais atributos são relevantes:

$$p(d, r) = N_m / N = \begin{bmatrix} m-d \\ r \end{bmatrix} \Big/ \begin{bmatrix} m \\ r \end{bmatrix} \qquad (r >= 1)$$

Esta probabilidade irá garantir que o análogo mais similar é a analogia mais apropriada.

3 Arquitetura Multiagente Improvisacional

Na arquitetura multiagente improvisacional proposta, os agentes são responsáveis pela criação e apresentação de algum conteúdo. A fim de realizar esta tarefa, eles assumem um dos três papéis: diretor, ator e diretor-ator. O diretor tem que coordenar os atores, informando aos mesmos seus cursos de ação, chamados *scripts*, e auxiliar os atores a resolver problemas inesperados. Os atores têm que seguir as instruções do diretor enquanto improvisam comportamentos apropriados a cada situação. O diretor-ator é um papel misto que os atores podem ter quando gerenciam ambas responsabilidades, de ator e de diretor. A arquitetura é improvisacional no sentido de que inclui improvisação tanto no ator quanto no diretor. Isto torna possível a incorporação de improvisação implícita (incluindo improvisação de texto e personalidade) relacionada a eventos previsíveis e conhecidos e improvisação explícita (significando improvisação para solução de problemas) relacionada a eventos conhecidos (através de experiências passadas) e não previstos no curso de ação atual do agente. Os atores e o diretor executam estes tipos de improvisação em diferentes níveis de abstração. O diretor está envolvido na fase preparatória da improvisação e o ator na fase de execução da improvisação.

Cada agente independente de seu papel está organizado em uma arquitetura de duas camadas. A camada de alto nível contém os processos relacionados às capacidades cognitivas dos agentes e a camada de baixo nível contém processos relacionados à percepção e ação no ambiente. A fim de suportar os dois tipos de improvisação mencionados anteriormente, a camada de alto nível é baseada nos processos do diretor improvisacional [10] e é composta por aquisição de conhecimento, construção das intenções e improvisação para solução de problemas, como mostra a figura 1.

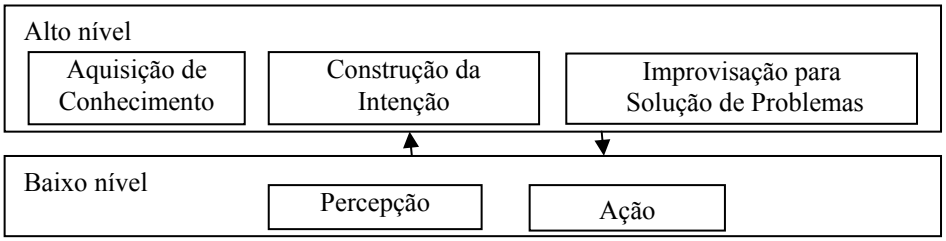


Figura 1. Arquitetura Multiagente Improvisacional

3.1 Intenção do Diretor

Para coordenar o comportamento dos atores, o diretor possui a sua própria intenção. O esquema da intenção do diretor pode ser visualizado na figura 2.

1. Executar processo de aquisição de conhecimento
2. Executar processo de construção das intenções para os atores
3. Enviar intenções (como scripts) para os atores
4. Esperar por uma percepção do ator solicitando ajuda
5. Enquanto a percepção é diferente de fim
   Início
   5.1 Executa módulo de improvisação para solução de problemas para gerar a operação improvisada requisitada pelo ator.
   5.2 Enviar a intenção (como script) reformulada para o ator
   5.3 Esperar por uma nova percepção
   Fim_Enquanto
6. Fim_Intenção

Figura 2: Intenção do Diretor

A improvisação implícita é realizada através dos módulos de aquisição de conhecimentos e construção da intenção, como satisfação de restrições. A improvisação explícita é realizada através do módulo de improvisação para solução de problemas, como analogia por similaridade. As próximas seções explicam o funcionamento dos módulos para o papel de diretor.

3.2 Módulo de Aquisição de Conhecimento

No processo de aquisição de conhecimento, o autor de uma apresentação fornece informação sobre a apresentação para o diretor. Esta informação pode ser uma seqüência de conteúdos que deve ser apresentada e falas relacionadas ao conteúdo. A seqüência pode ser informada em uma ordem completa ou parcial. Por exemplo, o autor pode dizer ao diretor que o ator tem que primeiro se apresentar dizendo alguma das seguintes falas: "*Olá, meu nome é Kira! Eu estou aqui para apresentar Porto Alegre para você*." ou "*Olá! Eu sou Kira e estou aqui para falar de Porto Alegre para você*.". Depois o ator pode escolher entre apresentar a história de Porto Alegre ou apresentar fatos sobre a localização de Porto Alegre. O autor também informa falas relacionadas a estes assuntos. A última ação é finalizar a apresentação dizendo uma das possíveis falas: "*Foi muito legal falar com você. Até uma próxima!*" ou "*Espero que tenha gostado da apresentação. Tchau!*". No exemplo anterior, o autor humano informou uma ordem parcial das ações que o ator deve executar. O autor humano fixou a primeira e última ação e o ator tem que escolher a ordem das ações intermediárias. A figura 3 demonstra como a informação segue no sentido da aquisição de conhecimento e construção da intenção.

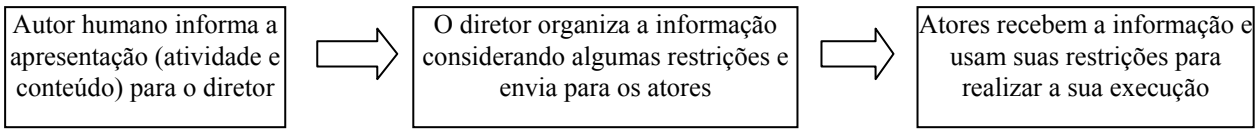


Figura 3: Fluxo de informação da aquisição de conhecimento para a construção da intenção

É importante observar que este é uma das maneiras na qual a informação flui. Em outros processos, o ator pode enviar informação para o diretor e o diretor par ao autor humano.

Como mencionado da seção 2.1, pode-se pensar nos componentes do planejamento tradicional como algo que os agentes podem usar para guiar seu curso de ação e não como algo que será usado para planejar todo o curso de ação com antecedência. Desta maneira, os componentes de pré-condição, ação e efeitos, informados por um autor de uma apresentação, podem ser vistos como envolvidos na organização de um tipo de apresentação. Após receber a informação de um autor humano, o diretor reconhece esta informação como uma estrutura parcialmente ordenada representando ações e a usa para construir as intenções dinâmicas dos agentes.

3.3 Módulo de Construção da Intenção

Baseado nas informações obtidas pelo módulo de aquisição de conhecimento, o diretor executa o seu módulo de construção de intenções para construir as intenções para os atores. As intenções também são chamadas de *scripts* abstratos de comportamento, pois elas são planos abstratos que guiam, mas não determinam o comportamento do agente. O módulo é dividido em dois submódulos responsáveis pelo escalonamento das atividades e escalonamento dos comportamentos que irão compor uma intenção do agente (figura 4).

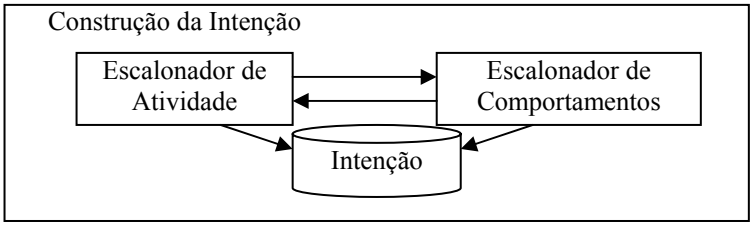


Figura 4: Módulo de Construção da Intenção

O módulo opera em um ciclo entre o escalonador de atividade e o escalonador de comportamento, que utilizam satisfação de restrições para construir a intenção do agente. O diretor opera sobre restrições de ordem para construir as intenções. Os tipos de restrições manipuladas são: precondições (indica as pré-condições relacionadas a ordenação de alguma execução); efeitos (que efeitos a execução de alguma atividade pode trazer e estado do script (que indica se o script está vazio ou não). A ativação do efeito dispara a satisfação de uma nova pré-condição, e assim o diretor vai construindo a intenção do agente. Alguns exemplos de restrições podem ser vistos na figura 5.

Ação	**Exemplo de Restrições**
• Procure por atividade cuja pré-condição é nenhuma	estado do script é igual a vazio
• Armazene um efeito intermediário, Armazene precedência no script, Chame o escalonador de comportamento e Armazene efeito no script	estado do script é diferente de vazio e existe atividade cuja precedência é igual ao efeito corrente

Figura 5: Exemplos de restrições utilizadas para construção da intenção

O escalonador de atividades é responsável por percorrer a estrutura de uma apresentação procurando por atividades que satisfazem as restrições de ordem apresentadas em um determinado momento. O escalonador de atividades funciona da seguinte maneira: procura por curso de ação igual ao objetivo informado; enquanto o efeito de uma atividade do curso de ação é diferente de fim, o diretor procura por atividades, que ainda não estejam na intenção, através de suas precedências. Ao encontrar, o diretor armazena a precedência na intenção, chama o escalonador de comportamento para armazenar a parte relativa à ação e armazena o efeito que será gerado após a execução da ação. Ao finalizar a construção da intenção como *script*, o diretor relaciona um ator ao *script* e grava a intenção em uma biblioteca de *scripts*.

O escalonador de comportamentos relaciona um conjunto de comportamentos possíveis para uma determinada atividade, considerando as restrições de comportamento que o agente possui. O ator, no momento da execução irá considerar suas restrições para escolher dentro do conjunto de comportamentos, o comportamento mais apropriado para uma determinada situação.

Através da interação entre o escalonador de atividades e o escalonador de comportamentos, é construída a intenção do agente como um conjunto de regras de decisão, que serão satisfeitas de acordo com a execução da intenção pelo ator. A estrutura da intenção pode ser visualizada na figura 6.

```
if <precedência1, ..., precedênciaN>
then  <conteúdo>
      <efeito>
```

Figura 6: Estrutura de uma intenção

Precedência é a precedência de alguma atividade. Conteúdo é uma indicação de um processo que deve ser chamado no ator para escolher que conteúdo e ação específica deve ser executado. O ator tem que usar suas restrições para escolher quais conteúdos e ações executar, porque o diretor somente informa alguma ordem e classes de ações para o ator. Efeito é o efeito ou efeitos relacionados à execução de uma atividade. A ativação de um efeito influencia na satisfação de uma ou mais precedências. Em alguns casos, pode existir mais de uma precedência que seja satisfeita em um determinado momento. Quando um ator está executando seu *script* e isto acontece, ele deve escolher qual é a melhor opção de acordo com as suas restrições (limitações e oportunidades). O ator tem que improvisar a execução do seu *script* considerando as restrições de uma determinada situação.

Como se pode observar, a intenção representada como um *script* abstrato de comportamento é uma descrição abstrata do que e de como o ator irá executar uma determinada atividade. O diretor não dita o comportamento do ator. Ele fornece informações gerais e deixa o ator livre para agir de acordo com as suas restrições. O diretor e o ator estão trabalhando em conjunto para apresentar algum conteúdo ao usuário.

3.4 Módulo de Improvisação de Solução de Problemas

Após adquirir conhecimento, construir intenções como *scripts* abstratos de comportamentos e enviá-las ao ator, o diretor espera por uma percepção. A estrutura da percepção é composta por: um tipo de percepção, identificação do agente, e uma descrição de objeto. Se o tipo de percepção é *falha*, a identificação do agente indica qual agente solicitou a solução de problema e qual intenção (*script* abstrato de comportamento) necessita de improvisação. A descrição de um objeto tem o mesmo conjunto de atributos de um objeto meio (ver seção 3.4.1). A ação de *falha* é gerada por um ator quando ocorre um evento que não foi previsto no seu curso de ação. Este evento pode gerar duas situações diferentes. Na primeira situação, o agente pode ter conhecimento que pode ser diretamente aplicável para resolver o problema. Desta maneira, ele poderia usar re-planejamento, embora algumas vezes esta não seja a solução mais apropriada, como mencionada anteriormente. No outro caso, o re-planejamento não é aplicável, pois o agente não tem conhecimento de como agir na situação. Consideramos que em ambos os casos, o problema pode ser resolvido por improvisação baseada em analogia por similaridade.

3.4.1 Objetos Fim e Objetos Meio

Para trabalhar com analogia por similaridade, assumimos que na analogia tanto os objetos fonte quanto os objetos alvo são descritos através de um modelo genérico de objetos. Neste modelo, os objetos são de dois tipos: fim e meio. Cada tipo de objeto é descrito usando um conjunto de atributos. Objetos classificados como fim são relacionados ao objetivo do agente, em um curso de ação específico. Objetos classificados como meio estão relacionados à maneira como um curso de ação específico é executado. O conjunto de atributos para objetos fim são: objetivo, efeito e características particulares. O conjunto de atributos para objetos meio são: usos possíveis e características particulares. O grupo de usos possíveis tem uma lista de usos para um objeto. O grupo de características particulares tem um conjunto de pares atributo-valor que especificam as características de um objeto. Os valores para os atributos (objetivo, efeito, lista de usos possíveis e características particulares) dependem do domínio da aplicação. Um objeto fim pode estar relacionado a objetos meio através das características particulares. Por exemplo, um objeto fim pode ter como característica particular o nome do objeto meio que poderia ser parte do seu curso de ação.

3.4.2 Tipos de Improvisação para Solução de Problemas

Improvisação para solução de problemas pode ocorrer de duas maneiras: usando um *meio improvisado* como uma maneira alternativa para alcançar o objetivo atual ou usando um *fim improvisado* como um objetivo alternativo para alcançar o objetivo atual através de algum efeito do objetivo alternativo que seja compatível com o objetivo atual. No primeiro caso se possui improvisação no nível de meios e no segundo caso, improvisação no nível de objetivos.

3.4.3 Arquitetura do Módulo de Improvisação para Solução de Problemas
A figura 7 mostra a arquitetura do módulo de improvisação para solução de problemas, com suas entradas e saídas. O módulo recebe como entrada a descrição de um objeto que provoca a falha, os objetos que representam o conhecimento do agente e a intenção que deve ser reconsiderada. A descrição de um objeto problema segue o mesmo padrão de um objeto do tipo meio, logo é composta por usos possíveis e características particulares. Baseado nestas entradas, o módulo executa a reconsideração da intenção usando um processo de analogia por similaridade. Para realizar isto, a arquitetura do módulo é composta de três submódulos: identificação do tipo de improvisação, construção da analogia e transformação da analogia em intenção.

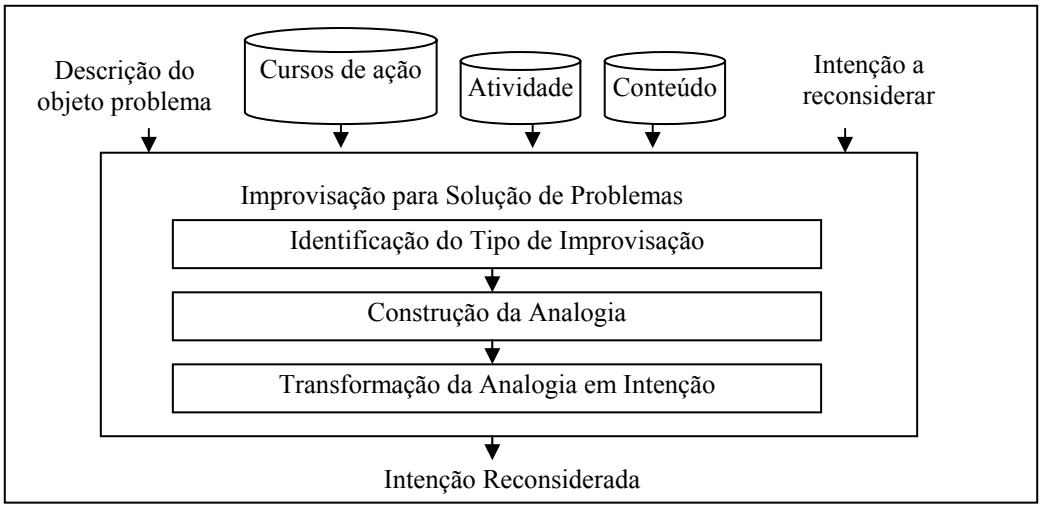


Figura 7: Módulos da Arquitetura de Improvisação para Solução de Problemas

3.4.3.1 Módulo de Identificação do Tipo de Improvisação
Este submódulo recebe a descrição do objeto que ocasionou o problema e envia para o submódulo de construção da analogia, o tipo de improvisação a ser executada, uma lista de possíveis fontes análogas e a descrição do objeto problema. Para identificar o tipo de improvisação a ser executado, o submódulo busca por objetos fim (na base de dados de curso de ação) que tem um objetivo compatível com um dos usos possíveis presentes na descrição do objeto problema. Se um ou mais objetos compatíveis são encontrados, o tipo de improvisação é de meio. Neste momento, o submódulo procura por objetos meio (na base de dados de conteúdos) que tenham usos possíveis e características particulares compatíveis com a descrição do objeto problema. Este processo resulta em uma lista de possíveis objetos análogos.

Se nenhum objeto fim é encontrado, o tipo de improvisação é de fim. Neste caso, o submódulo procura por objetos fim (na base de dados do curso de ação) que tenham efeito compatível com os usos possíveis da descrição do objeto problema, construindo uma lista de objetos análogos possíveis. Se esta lista é vazia, ou seja, não existe objeto fonte com efeitos compatíveis com os usos possíveis da descrição do objeto problema, existe o que chamamos de improvisação sem efeitos conhecidos. Neste caso, o submódulo procura por objetos meio (na base de dados de conteúdos) que tenham características particulares semelhantes às características particulares do objeto problema. Este processo constrói a lista de objetos análogos possíveis.

Desta maneira, o submódulo implementa improvisação de meio e dois tipos de improvisação de fim: com efeito conhecido e sem efeito conhecido. Tanto a improvisação de meio quanto a improvisação de fim com efeitos conhecidos são resultado de uma analogia forte, enquanto improvisação de meio sem efeito conhecido é resultado de uma analogia fraca. Analogia é chamada de fonte quando existe um relacionamento entre os usos possíveis da descrição do objeto problema e os efeitos dos objetos fim fonte. Analogia é chamada de fraca quando não existe tal relacionamento, mas existe alguma analogia entre as características particulares. Estes dois tipos de analogias sempre possibilitam ao agente produzir improvisações (achar uma solução para o problema), mesmo quando a analogia encontrada não é a mais adequada.

3.4.3.2 Módulo de Construção da Analogia
Este submódulo usa as idéias apresentadas por Russell [12] para selecionar o objeto análogo fonte mais adequado, dada a descrição de um objeto alvo. Deve-se definir a similaridade *s*, a diferença *d*, os atributos

relevantes *r* e a probabilidade *p(d,r)* para cada uma das fontes análogas encontradas pelo submódulo de identificação do tipo de improvisação. A similaridade *s*, os atributos alvo *m* e os atributos relevantes *r* são calculados para improvisação de meio e de objetivo sem efeito conhecido considerando o conjunto de atributos de características particulares e usos possíveis. Na improvisação com efeito conhecido somente o conjunto de atributos do tipo efeito é considerado. Isto ocorre porque quando um objetivo muda, as características particulares não necessariamente são compatíveis, mas os usos possíveis são compatíveis. Na improvisação de meio, o objetivo não muda, mas os meios podem ser diferentes do objetivo atual, desta maneira, o submódulo considera os usos possíveis para buscar por similaridades dentro dos objetos meio.

Após calcular a similaridade da fonte análoga, deve ser calculada a diferença entre os *m* atributos que descrevem o alvo e a similaridade *s*, como $d = m - s$. São assumido *r* atributos relevantes como um argumento de entrada e é calculada a probabilidade *p(d,r)* como a probabilidade de uma fonte *S*, diferindo do seu alvo em *d* atributos, combinar com os *r* atributos relevantes usando a fórmula apresentada na seção 2. A suposição de que não existe informação relevante, significa que todos os atributos são igualmente relevantes. O submódulo aplica a fórmula de probabilidade ($p(d,r) = N_m / N$) para cada um dos objetos análogos possíveis presentes no conjunto construído pelo submódulo de identificação do tipo de improvisação. O objeto análogo escolhido será aquele que tem a maior probabilidade de ser similar ao alvo. Ao final da sua execução, o submódulo envia para o submódulo que transforma a analogia em intenção o tipo de improvisação e a fonte análoga selecionada como mais similar.

3.4.3.3 Transformação da Analogia em Intenção
O submódulo recebe o tipo de improvisação e o objeto fonte que é a analogia mais apropriada e transforma esta analogia em intenção. Se o tipo de improvisação é de meio ou fim sem efeito conhecido, a intenção a ser desenvolvida é uma atualização da intenção atual. Esta atualização irá conter o objeto fonte mais similar ao objeto que ocasionou o problema. Se o tipo de improvisação é de fim com efeito conhecido, o submódulo constrói uma nova intenção baseada no objeto fim escolhido. Como mencionado anteriormente, um objeto fim tem um relacionamento com objetos meio através das suas características particulares. O submódulo ativa o módulo de construção da intenção, para que iniciando do objeto fim e dos seus relacionamentos a objetos meio, construa uma nova intenção e envie-a de volta para o submódulo de transformação da analogia em intenção.

4 Estudo de Caso: Museu Virtual SAGRES

O museu virtual SAGRES é um sistema que visa suportar o processo de aprendizagem através da interação entre visitantes e entre um visitante e os recursos do museu [23]. O SAGRES facilita a organização de visitas ao museu, apresentando as bases de informações do museu de maneira adaptada às características dos visitantes. A arquitetura proposta está sendo usada para criar guias virtuais e assistentes pessoais no SAGRES. Até o momento, realizamos três simulações do funcionamento da arquitetura para a apresentação de três conteúdos que não foram previstos no curso de ação inicial do agente. Analisamos em detalhe o módulo de improvisação de solução de problemas para os diferentes tipos de improvisação (meio, fim com objetivo conhecido e de fim sem objetivo conhecido) e verificamos que o módulo produz respostas coerentes para os testes aplicados, sempre escolhendo o objeto fonte análogo que corresponde à analogia mais similar.

Como considerado na seção 3.4.3.1, a improvisação de fim sem efeito conhecido pode produzir uma analogia fraca e isto aconteceu em uma das simulações. Embora a resposta produzida não fosse diretamente relacionada ao conteúdo solicitado, existia um relacionamento entre as características particulares do conteúdo, indicando que eles tem um mesmo super conjunto de conteúdos. Isto mostra que mesmo que os agentes não tenham conhecimento completo do conteúdo, eles podem produzir respostas plausíveis, prevenindo a sua falha de execução. Neste caso, os agentes podem expressar sua falta de conhecimento específico explicando que eles não tem conhecimento do conteúdo, mas conhecem um outro que é semelhante, em certas características, ao que foi requisitado. Fazendo isto, os agentes podem agir como os humanos agem na mesma situação, mostrando uma analogia a uma situação conhecida e apresentando um comportamento credível e esperto, produzindo a ilusão de vida tão desejada nos agentes computacionais.

5 Considerações Finais

A arquitetura multiagente improvisacional se baseia em uma arquitetura BDI tradicional [24] e pode ser mapeada para uma arquitetura BDI diagramática como apresentada em Wooldridge [25] com uma extensão. O processo de aquisição de conhecimento representa a função de revisão de crenças. A construção da intenção contém as funções de filtro e opções. Percepção representa o sensor de entradas e ação representa a função de ação. A extensão está presente no processo de improvisação para solução de problemas que contém as

características das funções opções e filtro para construir um curso de ação alternativo através de analogia por similaridade quando algo inesperado acontece por acidente e o agente não sabe como resolvê-lo.

Mostrar o mapeamento entre a arquitetura proposta e a arquitetura BDI diagramática de Wooldridge é importante a fim de inserir improvisação como uma característica fundamental de agentes racionais, aperfeiçoando suas capacidades de tratar problemas que não foram antecipados no planejamento do seu curso de ação. A escolha da arquitetura BDI para mostrar a possibilidade de usar improvisação em agentes racionais foi realizada devido à significância e reconhecimento do BDI para descrever comportamento de agentes racionais. Este mapeamento supera o vazio existente entre improvisação e inteligência artificial e mostra que improvisação é uma técnica natural de resolução de problemas.

Neste artigo apresentamos uma abordagem para incluir improvisação no processo de resolução de problemas em agentes racionais, considerando analogia e raciocínio baseado em casos como sinônimos. A analogia por similaridade foi escolhida porque acreditamos que ela é idéia mais próxima à improvisação. Improvisação necessita de respostas rápidas para situações inesperadas e usa os recursos disponíveis para produzir estas respostas. A analogia por similaridade fornece respostas rápidas [12] e utiliza experiências passada (recursos) para produzir novas soluções para problemas inesperados.

Com o módulo de improvisação de solução de problemas estendemos os trabalhos anteriormente realizados em improvisação, propondo uma abordagem para tratar eventos inesperados através de um agente diretor. Trabalhos anteriores somente consideraram agentes atores capazes de tratar eventos relacionados a situações esperadas através de planejamento e improvisação implícita. Estendemos o uso da improvisação para a geração de comportamento improvisado em resposta a situação inesperada que pode ocorrer quando o agente está executando o seu curso de ação. A habilidade de produzir respostas rápidas e coerentes para situações inesperadas é fundamental para os agentes, especialmente aqueles responsáveis pela interação com usuário, e o uso de improvisação na solução de problemas aperfeiçoa esta habilidade.

As simulações executadas no SAGRES mostram que o módulo de improvisação para solução de problemas pode produzir respostas relevantes para eventos não previstos. Contudo, com estas simulações não é possível realizar considerações sobre a eficiência do módulo. Estamos realizando novos testes a fim de comparar a performance da arquitetura de guias virtuais com e sem o módulo de improvisação e observar resultados mais completos e adequados em relação à capacidade do agente em resolver problemas inesperados.

Referências

[1] Pollack, M. E. The use of plans. Artificial Intelligence, Vol. 57. Elsevier Science Publishers (1992) 43-68

[2] Russell, S.; Norvig, P. Artificial Intelligence: a modern approach. Upper Saddle River. (2003)

[3] Hayes-Roth, B., Doyle, P. Animated Characters. In: Autonomous Agents and Multi-Agent Systems, Vol. 1, Kluwer Academic Publishers, (1998) 195-230

[4] Ball, G.; Ling, D.; Kurlander, D.; Miller, J.; Pugh, D.; Skelly, T.; Stankosky, A.; Thiel, D.; Van Dantzich, M.; Wax, T.: Lifelike Computer Characters: The Persona Project at Microsoft. In: Software Agents. Menlo Park, California: AAAI Press. (1997)

[5] Koda, T. Agents with Faces: A Study on the Effects of Personification of Software Agents. Master Thesis, MIT Program in Media Arts and Sciences. (1996)

[6] Loyall, B.: Believable Agents: Building Interactive Personalities. PhD Thesis Carnegie Mellon University, Technical Report CMU-CS-97-123. (1997)

[7] Hayes-Roth, B.; Browston, L.; Sincoff, E. Directed Improvisation by Computer Characters. Technical Report KSL-95-04 – Stanford University. (1995)

[8] Bates, J. The Nature of Characters in Interactive Worlds and The OZ Project. Technical Report CMU-CS-92-200 – Carnegie Mellon University. (1992)

[9] Perlin, K.; Golberg, A. Improv: A System for Scripting Interactive Actors in Virtual Worlds. In: Computer Graphics, vol. 29. (1996)

[10] Spolin, V. Improvisation for the Theater: A Handbook of Teaching and Directing Techniques. First Edition. Nothwestern University Press. (1963)

[11] Chacra, S.: Natureza e Sentido da Improvisação Teatral. Editora Perspectiva. (1983)

[12] Russell, S.: Analogy by Similarity. In David Helman (Ed.), Analogical Reasoning, Boston, MA: D. Reidel. (1988)

[13] Agre, P. E. The Dynamic Structure of Everyday Life. Phd Thesis, MIT Artificial Intelligence Laboratory, Technical Report 1085 (1988)

[14] Agre, P. E.: Computation and Human Experience. Cambridge University Press (1997)

[15] Agre, P. E., Chapman, D.: Pengi: An Implementation of a Theory of Activity. Sixth National Conference on Artificial Intelligence. Morgan Kaufmann Publishers. Vol. (1987) 268-272

[16] Anderson, J. E.: Constraint Directed Improvisation for Everyday Activities. Phd Thesis University of Manitoba. (1995)

[17] Suchman, L. A.: Plans and Situated Actions: The problem of human machine communication. Cambridge: Cambridge University Press (1987)

[18] Pfleger, K., Hayes-Roth, B.: Plans Should Abstractly Describe Intended Behavior. In Alex Meystel, Jim Albus, and R. Quintero (eds.): Intelligent Systems: A Semiotic Perspective, Proceedings of the 1996 International Multidisciplinary Conference, Vol. 1 (1996) 29-34.

[19] Marriott, K., Stuckey, P. J.: Programming with Constraints: An Introduction. MIT Press, Cambridge Massachusets (1998)

[20] Stefik, M.: Introduction to Knowledge Systems. Morgan Kaufmann Publishers, Inc. San Francisco (1995)

[21] Russell, S.: A Quantitative Analysis of Analogy by Similarity. In Proceedings of the Fifth National Conference on Artificial Intelligence, Philadelphia, PA: Morgan Kaufmann. (1986)

[22] Davies, T. R; Russell, S.: A Logical Approach to Reasoning by Analogy. In Proceedings of the Tenth International Joint Conference on Artificial Intelligence, Milan, Italy: Morgan Kaufmann,. (1987)

[23] Bertoletti, A. C.; Costa, A. C. R. SAGRES – A Virtual Museum. In: Museums and The Web 1999 Conference. USA: New Orleans, Louisiana. (1999)

[24] Bratman, Michael E.; Israel, David J.; Pollack, Martha E. Plans and Resource-Bounded Practical Reasoning. Computational Intelligence, 4 (4). (1988) 349-355

[25] Wooldridge, Michael. Intelligent Agents. In: Multiagent systems: A Modern Approach to Distributed Artificial Intelligence. Gerhard Weiss (editor). The MIT Press, Cambridge, Massachusetts. (1999) 27-77

Revealing Undercover Refinement in UML Modeling

C. Pons[1], G. Perez[1], and R-D Kutsche[2]

[1] LIFIA – Laboratorio de Investigación y Formación en Informática Avanzada, University of La Plata, 1900 Buenos Aires, Argentina
cpons@info.unlp.edu.ar
[2]*CIS Computation and Information Structures CIS,* Technical University of Berlin, Faculty IV, Berlin, Germany

ABSTRACT. Although the Abstraction artifact allows for the explicit documentation of the abstraction/refinement relationship in UML models, an important amount of variations of this relationship remains unspecified, in general hidden under other notations. The starting point to enable traceability of requirements across refinement steps is to discover and precisely capture the various forms of the abstraction/refinement relationship, in particular those forms which are hidden in the model. In this article we formally describe a number of undercover refinements and present PAMPERO, a tool integrated in the Eclipse environment, based on the formal definition of refinement. The tool supports the documentation of explicit refinements and the semi-automatic discovering and documentation of hidden refinements.

Keywords: UML, refinement, abstraction, traceability, CASE tool.

1. Introduction

Abstraction [Dijkstra, 76] is the key to mastering complexity. Abstraction facilitates the understanding of complex systems by dealing with the major issues before getting involved in the detail. Apart from enabling for complexity management, the inverse of abstraction, refinement, captures the essential relationship between specification and implementation. Refinement relationship makes it possible to understand how each business goal relates to each system requirement and how each requirement relates to each facet of the design and ultimately to each line of the code.

Documenting the refinement relationship between these layers allows developers to verify whether the code meets its specification or not, trace the impact of changes in the business goals and execute test assertions written in terms of abstract model's vocabulary by translating them to the concrete model's vocabulary.

The standard modeling language UML [OMG, 2001] provides an artifact named *Abstraction* (a kind of Dependency) to explicitly specify abstraction/refinement relationship between UML model elements. In the UML metamodel an Abstraction is a directed relationship from a *client* (or clients) to a *supplier* (or suppliers) stating that the client (the refinement) is dependent on the supplier (the abstraction). The Abstraction artifact has a meta attribute called *mapping* designated to record the abstraction/implementation mappings, that is an explicit documentation of how the properties of an abstract element are mapped to its refined versions, and on the opposite direction, how concrete elements can be simplified to fit an abstract definition. The more formal the mapping is formulated, the more traceable across refinement steps the requirements are.

Although the Abstraction artifact allows for the explicit documentation of the abstraction/refinement relationship in UML models, an important amount of variations of abstraction/refinement remains unspecified, in general hidden under other notations. For example UML artifacts such as generalization, composite association, use case inclusion, among others, implicitly define abstraction/refinement relationship. The starting point to enable traceability of requirements across refinement steps is to discover and precisely capture the various forms of the abstraction/refinement relationship, in particular those forms which are hidden in the model.

To experiment, we created a tool integrated in the Eclipse environment [IBM, 2003], called PAMPERO (**P**recise **A**ssistant for the **M**odeling **P**rocess in an **E**nvironment **R**efinement **O**riented), based on the formal definition of refinement. The tool supports the documentation of explicit refinements (i.e. Abstractions artifacts with their corresponding mapping expressions) and the semi-automatic discovering and documentation of hidden refinements.

In the remainder of this article we will describe a number of undercover refinements in UML modeling. Refinement can be established between model elements of either the same kind (e.g. between two classes) or different kind (e.g. between a use case model and a collaboration model) [Pons et al. 2000; Giandini and Pons, 2002; Pons et al.,2003]. In this article we restrict our attention to relationships between model elements of the same kind, in particular we focus on three UML artifacts: Classes, which are described in section 2; Associations presented in section 3 and Use Cases which are analyzed in section 4. For each one of these artifacts the discussion

comprises two dimensions: intension and extension, where the intension of a modeling artifact is equated with its definition or specification, while its extension refers to the set of elements that fall under that definition.

2. Class Refinement

Classes serve as specifications for the properties of sets of objects that can be treated alike. The intension of a Class is defined as a pair (Attr, Ops) where Attr={a_1,...a_n} is a set of Attribute's description and Ops= {op_1,...op_m} is a set of Operation's description.

Figure 1a shows the UML artifact specifying abstraction/refinement relationship between Classes. The Catalysis methodology (d'Souza and Wills, 1998) mentions that refinement between Classes (or Types) can be realized in two different ways:

a) *Attribute (or model) Refinement*: The refined Class, B, is obtained by adding a new attribute, $attr_k$, to the abstract Class A. That is to say, B = A + ({$attr_k$}, {})[1]. Other case takes place when the Class B is obtained from Class A=({a_1, ..., a_k,..., a_n} , Ops) by replacing an attribute a_k by its refinement, that can be one or more new attributes, a_{k1}, ..., a_{kl}. That is to say, B= A[a_k | a_{k1}, ..., a_{kl}][2]. For example, figure 1b shows that the attribute *length* in Class *Segment* is refined by the attributes *xinitial* and *xfinal* through the mapping <length = xfinal – xinitial>.

b) *Operation Refinement*: The refined Class, B, is obtained by adding a new operation, op_k, to the abstract Class A. That is to say, B = A + ({}, {op_k}). On the other hand the Class B can be obtained from Class A=(Attr, {op_1, ..., op_k,..., op_n}) by replacing an operation op_k by its refinement, that can be one or more new operations, op_{k1}, ..., op_{kl}. That is to say, B= A[op_k | op_{k1}, ..., op_{kl}]. For example, figure 1c shows that the operation *stretch* in Class *Segment* is refined by the operations *moveXini* and *moveXfin* through the mapping <stretch(w) = moveXini(-w/2) ; moveXfin(w/2)>.

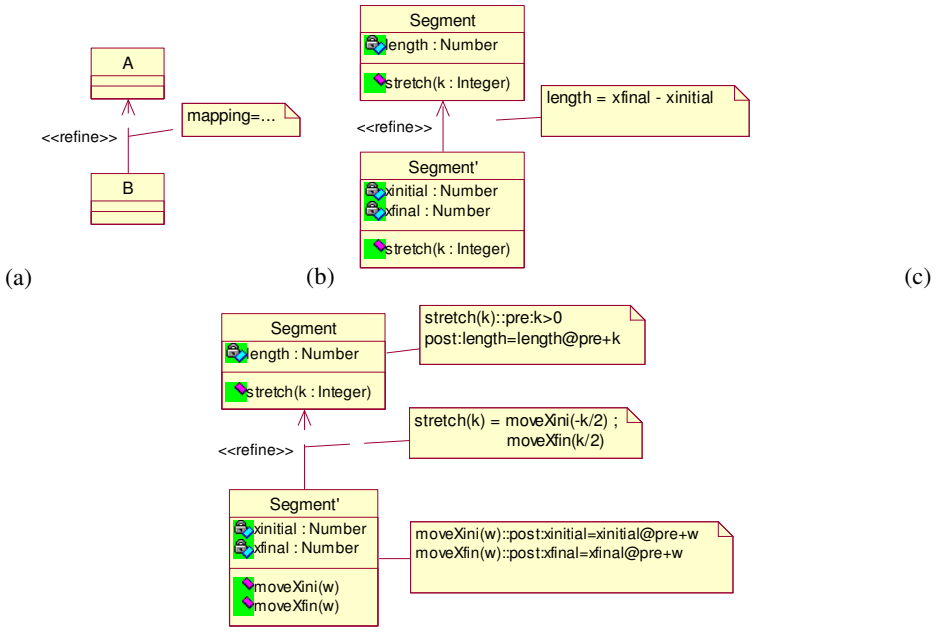


Figure 1: Abstraction/Refinement relationship between Classes. (a) UML notation. (b) Attribute refinement. (c) Operation refinement.

A refined Class is specified in a language that is richer than the language of its abstraction. However, there exists always a mapping from the abstract to the refined language (figure 2a), usually called "implementation mapping" [Cardelli and Wegner, 1985]. This mapping makes it possible to translate OCL expressions written in the abstract language to the refined language, for example, the OCL expression (**Context** Segment **inv** self.length > 0) can be translated to the expression (**Context** Segment' **inv** self.xfinal – self.xinitial > 0).

[1] Addition of Class intension is defined in the usual way: (A,O) + (A',O') = (A∪A', O∪O').

[2] B[y | x_1,...,x_n] denotes the replacement of element y in B by elements x_1,.., x_n.

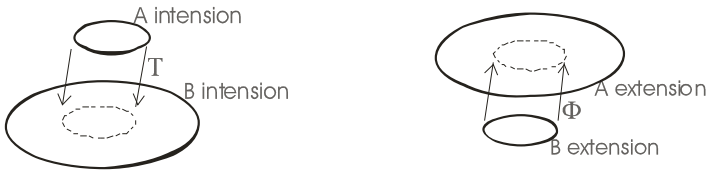


Figure 2: Mapping domains. (a) Implementation mapping. (b) Abstraction mapping .

On the semantics side, the extension of a Class is the set of objects meeting its definition:

extension: Class → Set (Object)

The semantics counterpart of the "implementation mapping" is the "abstraction mapping" that transforms each refined object to its abstract representation (figure 2b). For example, in the view of Objects as labeled records [Abadi and Cardelli, 1996], the mapping attached to the Abstraction artifact in figure 1b, can be defined in the following way: Φ : Record → Record,

Φ ([xinitial=x, xfinal=y, stretch=m]) = [length=y-x, stretch=m]

In all cases, the refinement is a more constrained specification, meaning that all properties specified for an abstraction when translated to its refined language also hold for its refinements, while more properties may hold for the refinement. Therefore the refinement is satisfied by a reduced number of objects. The meaning of Class refinement is that the extension of an abstraction (the supplier) includes the abstract representation of the extensions of all its refinements (the clients):

Context d:Abstraction **inv**:
d.supplier.extension.*includes*(d.client.extension.collect(d.abstractionMapping))

So far we have described the ways to explicitly define Abstraction/Refinement relationship between Classes, however there are other cases that remains hidden and should be discovered in order to allow us to formally check the refinement relationship and to trace properties from the abstract to the concrete models and backwards. In the remainder of this section we describe a number of forms of undercover refinement.

2.1 Refining by Specialization

The technique of generalization/specialization [Aristotle], which goes hand in hand with Inheritance [Booch, 91] [Wegner and Zdonik, 88], is a central issue in the object oriented paradigm. It is applied to enable reuse, so that less effort is spent when we re-specify things that have already been specified in a more abstract or more general way. In the object oriented paradigm a Class describes the structure and behavior of a set of objects, however it does so incrementally by describing extensions (increments) to previously defined classes (its parents or superclasses).

Figure 3 shows the syntactical connection between Generalization and Abstraction. The UML Generalization artifact (Figure 3a) relates two classes: the parent and the child. The child is not a self contained model, it is just an increment. While, on the other hand the Abstraction relationship (Figure 3b) relates self contained models that are obtained by combining the superclass with the increment.

Two cases of specialization can be distinguished: *Specialization without overriding* (the subclass adds new features without intersection with features in the superclass) and *Specialization with method overriding* (the subclass refines a method of the superclass). We do not consider the case of arbitrary method redefinition, only method refinement where the abstract version is replaced by the refined version of the method.

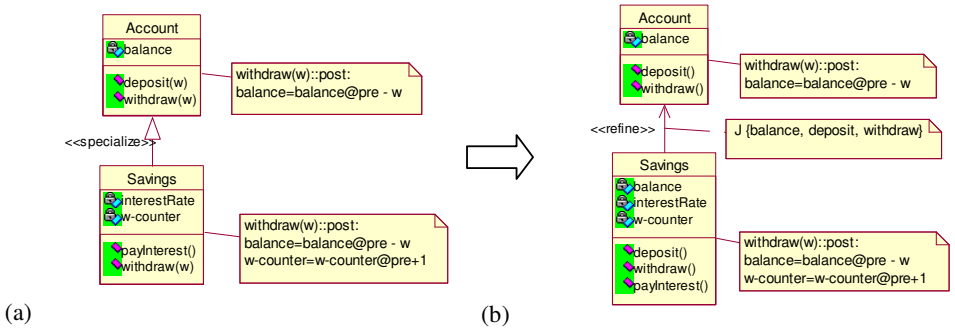
We define the mapping, reveal: Generalization → Abstraction, with the goal of making up the Abstraction artifact which is hidden under each Generalization artifact. In both cases the refinement is obtained by combining the parent and the child intension (considering method overriding if it is present).

Context Generalization **def**: reveal(): Abstraction
pre: self.parent.oclIsKindOf(Class) and self.child.oclIsKindOf(Class)
post: result.stereotype=<<refine>> and
result.supplier = self.parent and
result.client = (self.parent ⊕ self.child)[3] and

[3] Addition of class intension with method overriding is defined in the following way,

result.implementationMapping= id
result.abstractionMapping(r) = r⌋ Σ, [4]where Σ= self.parent.allFeatures

The implementation mapping is the identity function (even in the presence of overriding, abstract expressions are mapped to themselves because overriding preserves signatures). The abstraction mapping returns an abstract version of the refined object by keeping only the features defined in the parent Class while forgetting the rest.



(a) (b)

Figure 3: Refinement hidden under Specialization: (a) Generalization/Specialization relationship. (b) Abstraction/Refinement relationship derived from the Generalization.

On the semantics side, the subclass introduces a differentiation between the objects specified by the superclass. That is to say, as a consequence of adding more and more detail to the description of a class, a new characteristic that it is not present in all the individuals described by the class, is revealed. Different subsets have different characteristics. Consequently, the specialization technique allows modelers to implicitly specify refinement relationship that generates a partition of the class's extension into two or more subsets. For example, let d be the Abstraction artifact derived from the specialization of Class Account into two subclasses, Savings and Checking:

Account.extension = Savings.extension.collect(d.abstractionMapping).
union(Checking.extension.collect(d.abstractionMapping))

2.2 Refining by Decomposition

Generally, things are composed by smaller things, and this recursively. Composition is a form of abstraction: the composite represents its components in sufficient detail in all contexts in which the fact of being composed is not relevant. However, UML (and o-o modeling in general) has no notion of composition as a form of model abstraction. Consequently, the abstraction relationship remains hidden under composite association relationship. Figure 4a shows an example, where Account is a composite object holding a history of Movements. Additionaly the class Account has a basic attribute called initialBalance storing the value of the balance at the beginning of a period, and a derived attribute recording the current balance. Finally, there is an OCL constraint specifying how the current balance is calculated.

In [Steimann et al., 2003] it is observed that composite association, such as the one in figure 4a, is not a model abstraction relationship, which is reflected in the fact that Movement.extension is not included into Account.extension (In fact, there is a type mismatch between these two extension sets). Composite association is a relationship at the instance level (figure 4); instances of Account are composed by instances of Movement.

Let A=(Attr, Ops) and A'=(Attr', Ops') and Base= Ops.reject(op| Ops'.collect(name).includes(op.name)),
A ⊕ A' = (Attr ∪ Attr' , Base ∪ Ops')

[4] The restriction functor ⌋ Σ takes a labeled record and returns a restricted version of the record containing only those labels that are defined in Σ. For example: [balance=b, interesRate=r, w-counter=c, deposit=S1, withdraw=S2, payInterest=S3] ⌋ {balance, deposit, withdraw} = [balance=b, deposit=S1, withdraw=S2].

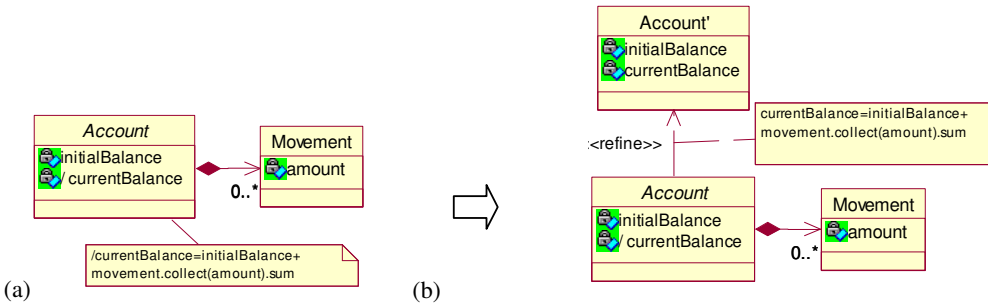


Figure 4: Refinement hidden under decomposition: (a) Composite Association relationship. (b) Abstraction/Refinement relationship derived from the Composite.

However from a composite associations we can derive a model abstraction relationship called abstractions by composition (see figure 5). In this case the relationship is established between models instead of being established between instances; for example, the Class Account' in figure 4b is an abstraction of the Class Account. Conversely, Account is a refined version of Account', showing more details (i.e. the fact of being a composite).

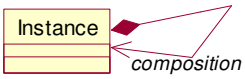


Figure 4: Composite at extension level.

Figure 5: Composite at intension level.

We define a mapping, reveal: Association → Abstraction, that returns the Abstraction artifact which is derived from each Composite Association artifact:

context Association **def**: reveal() : Abstraction
pre: self.connection.select(e| e.aggregation=#composite).size=1--the association is a composite-- and
self.connection.forAll (e|e.type.oclIsKindOf(Class)) -- the association connects classes --
post: result.stereotype=<<refine>> and
result.supplier = self.compound.hideParts and
result.client = self.compound

context Association **def**: compound (): Classifier -- returns the composite participant of the association--
post: result =self.connection.detect(e| e.aggregation=#composite).type

context Classifier **def**: hideParts(): Classifier -- returns an abstraction of the classifier by hiding its parts--

Neither implementation nor abstraction functions can be automatically derived from the composite, because more than one design decision can apply. However some hints are provided automatically. In the example in figure 4 the specification of the derived attribute currentBalance is suggested as implementation mapping making it possible to translate OCL invariants such as (**Context** Account' **inv** currentBalance>0) to a refined version (**Context** Account **inv** initialBalance+movements.collect(amount).sum > 0).

Regarding the extension sets, if d is an Abstraction resulting from a composite Association, then
d.supplier.extension = d.client.extension.collect(d.abstractionMapping), where the abstraction mapping transforms each composite object to its abstract representation. For example, the abstraction mapping in figure 4b is defined in the following way:

d.abstractionMapping ([initialBalance=b, currentBalance=f, movements=s]) =
[initialbalance= b, currentBalance= b + s.collect(amount).sum]

2.3 Other cases

Other UML constructs hiding Class refinement are the Interface dependency, the Instantiation relationship, the Parameterization construct, among others. Due to space limitation we cannot explain here all these cases, but the results are similar to the ones presented above.

3. Association Refinement

The UML provides an artifact named Association to specify relationships between instances of Classifiers. The association's intension contains the declaration of the types involved in the association, as well as other properties such as navigability, multiplicity, etc.

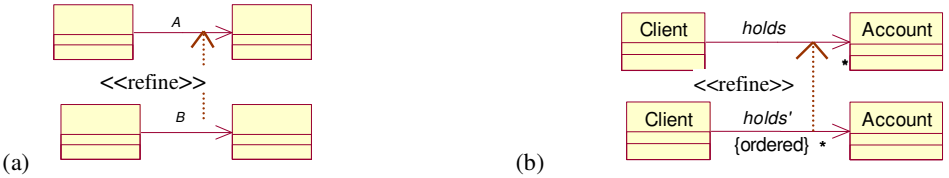


Figure 5: UML Abstraction/Refinement relationship between Associations. (a) UML notation. (b) Association refinement by constraining properties.

During the development process both classes and its relationships are gradually refined. It is usual to zoom in or out in both dimensions at the same time. Figure 5a shows the UML artifact to specify Abstraction/Refinement relationship between Associations. A form of explicit refinement takes place when an abstract association is constrained in some way, for example disallowing navigability, adding the constraint that the elements should be ordered, etc. Conversely, the abstraction is obtained by relaxing properties from the refinement. Figure 5b displays an example, where holds' is a refinement of holds.

Concerning the extension set, Associations are interpreted as relations in the mathematical sense, i.e., as subsets of the Cartesian products of the extensions of the involved types. The extensions of an association are sets of tuples called links:

extension: Association -> Set (AssociationInstance)
AssociationInstance = Set(Link)

The refinement is a more constrained specification, therefore it is satisfied by a reduced number of instances:

Context d:Abstraction **inv**:
d.supplier.extension.*includes*(d.client.extension.collect(d.abstractionMapping))

Refinement between Association can be explicitly specified, as described above; however there are other cases that remains hidden and should be discovered and made explicit. The different forms of undercover Abstraction/Refinement relationship between Associations are analyzed in the remainder of this section.

3.1 Refining by Specialization

Consider that a specialization is applied to the class Account generating two classes: Savings and Checking. At the same time, a specialization is applied to Client and a new class comes into existence: YoungClient. In this domain YoungClients are allowed to open only Savings while the remaining Clients may hold both Checking and Savings accounts (see figure 6a).

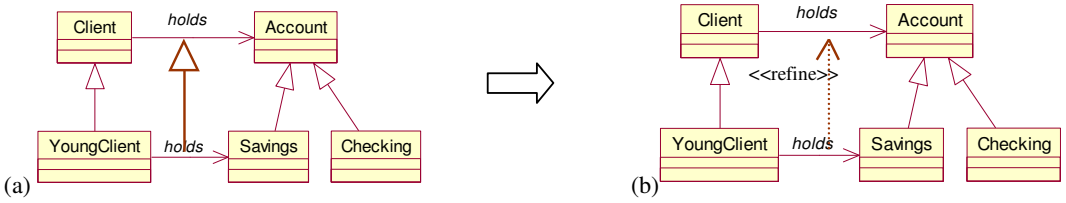


Figure 6: Association refinement hidden under a sub classification: (a) UML Generalization/Specialization relationship. (b) UML Abstraction/Refinement relationship.

Generalization between Associations is subtly different from Generalization between other UML artifacts. While generalization hierarchies in general reflect incremental development, generalization hierarchy of association is not incremental, but restrictive. The child Association frequently is a self contained model not just an increment, adding some constraints (e.g. restricting AssociationEnd´s type, disallowing navigability, adding the constraint that the elements should be ordered)

Figure 6 shows the syntactical connection between Generalization and Abstraction. The UML Generalization artifact in figure 6a relates two associations: the parent and the child. In this case both are self contained models. Therefore, the Generalization artifact can be simply replaced for an Abstraction artifact, see figure 6b, without altering the model's meaning.

The mapping reveal: Generalization → Abstraction, returns the Abstraction hidden under the Generalization.

context Generalization **def**: reveal() : Abstraction
pre: self.parent.oclIsKindOf(Association) and self.child.oclIsKindOf(Association)
post: result.supplier = self.parent and result.client = self.child and
result.stereotype=<<refine>>

Semantically, the generalization/specialization relationship between associations denotes an inclusion relation between the corresponding extension sets. Additionally, this form of refinement splits the set of abstract links into two or more subsets. For example, the association Client-*holds*-Account is partitioned into two refined sub-associations, as follows: Client-*holds*-Account.extension = YoungClient-*holds*-Savings.extension.union(ElderClient-*holds*-Account.extension), where ElderClient denotes the set of Clients who are not Young clients. Account abstractly denotes the union of both Savings and Checking.

3.2 Refining by Link Decomposition

A group of semantically related associations can be subsumed by an abstract association. For example Figure 7 shows two association between Client and Bank: *holdsAccount* and *hasCredit*. These two associations can be abstracted in a single association, called *worksWith*.

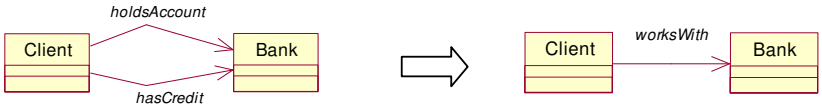


Figure 7: association abstraction.

In the UML Associations cannot be compound (only Classifiers can be compound). However, other languages, such as the one proposed by Catalysis [D'Souza and Wills, 1998], enable us to specify composition of Associations; see figure 8a.

Composition of association is neither a form of model abstraction in the UML. You could feel tempted to use the Abstraction artifact to specify composition of Associations, but to declare that *hasCredit* is a refinement of *worksWith* is as wrong as saying that Movement is a refinement of Account, in model in figure 3, despite the fact that here the type mismatch is not so evident.

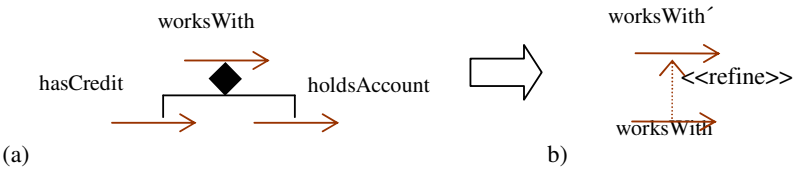


Figure 8: Association refinement hidden under link decomposition: (a) Composite/Component relationship in Catalysis. (b) Abstraction/Refinement relationship derived from the Composite.

In parallel direction to the composition between classes, the composition between associations is a relation at the instance level, specifying that each link of the compound association is composed by links of the component associations (figure 9). For example, let (c1,b1) be a link belonging to the extension of worksWith association, meaning that a Client c1 works with a Bank b1. We may zoom into this link revealing that the working relationship between c1 and b1 actually embraces two relationships: c1 has a credit in b1 and c1 holds an account in b1. That is to say, the link (c1,b1) of the compound association contains links of the component associations.

Figure 9: composition at the extension level.

Figure 10: composition at the intension level.

Once again, from a composition we can derive a model abstraction relationship called abstractions by composition (see figure 10). In this case the relationship is established between models instead of being established between instances. For example, the Association worksWith´ is an abstraction of the Association worksWith (see figure 8b). Conversely, worksWith is a refined version of worksWith´, showing the fact of being composed by two sub-associations.

The mapping, reveal: Association → Abstraction, returns the Abstraction artifact which can be derived from each decomposition of Association. Assuming that the UML metamodel is modified in order to permit the definition of Associations between Associations, the mapping is formally defined in the following way:

context Association **def**: reveal() : Abstraction
pre: self.connection.select(el e.aggregation=#composite).size=1--the association is a composite-- and
self.connection.forAll (ele.type.oclIsKindOf(Association)) -- the association connects associations--
post: result.stereotype=<<refine>> and
result.supplier = self.compound.hideParts and result.client = self.compound

context Association **def**: hideParts(): Association -- returns an abstraction of the association by hiding its parts--

4. Use Case Refinement

The Use Case construct is used to define the behavior of a system or other entity without revealing the entity's internal state. The intension of a Use Case consists of a pair (Attr, Ops) where Attr=(a_1,...a_n) is a set of Attribute's description and Ops= (op_1,...op_m) is a set of Operation's description. To simply we consider only one operation because that is the usual case. Operation is defined by pre and post conditions.

The extension of a Use Case is a sequence of actions that the entity can perform interacting with actors of the system, such that if the precondition holds, after executing the sequence of actions, the post condition is ensured:

extension: Use Case → Set (Scenario)
Scenario = Seq(Action)
uc.*extension* = {<a_1,…,a_n> | uc.*pre*{<a_1,…,a_n>}uc.*post* }[5]

Use Case refinement is obtained by refining attributes and/or by constraining the operation specification. Therefore the refinement is satisfied by a reduced number of scenarios. The meaning of use case refinement is that the extension of an abstraction includes the abstract representation of the extensions of all its refinements:

Context d:Abstraction **inv**:
d.supplier.extension.*includes*(d.client.extension.collect(d.abstractionMapping))

In the following sections we analyze some forms of hidden Abstraction/Refinement relationship between Use Cases.

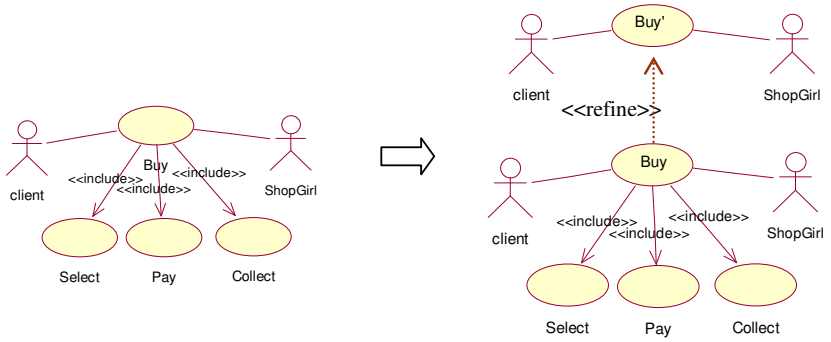
4.1 Refining by Action Decomposition

Action abstraction is the technique of treating an interaction between several participants as one single action. Then it is possible to zoom into, or refine, an action to see more detail. What was one single action is now seen to be composed of several actions. Each one of these actions can be split again into smaller ones, into as much detail as required.

This form of abstraction remains hidden because of the fact that UML does not consider composition as a form of model abstraction. To specify composite actions UML provides a relationship between Use Cases called *Include*. Figure 11a shows an example, where Buy is a composite action holding three constituent parts: Select, Pay and Collect .

[5] Here < a_1,…,a_n > stands for a sequence of actions executable in some way. Using the formalism of Hoare´s logic we say that a sequence of actions S is correct with respect to precondition p and postcondition q (denoted p{S}q), when starting in any state that satisfies precondition p, the actions terminates in a state satisfying q. Function *pre* (respectively *post*) returns the precondition (respectively postcondition) of the (only) operation of the use case.

In the abstract model the action Buy is treated as a single action whereas the refined model shows that the action Buy is composed by three sub actions. It should be observed that Use Case inclusion, such as the one in figure 11a, is not a model abstraction relationship, but a relationship at the instance level (figure 12). It specifies that each UseCaseInstance of the compound Use Case is composed by UseCaseInstances of the component Use Cases For example, let <Ana_buys_a_dress> be a UseCaseInstance belonging to the extension of the Buy Use Case. We may refine this instance revealing that actually it includes three instances inside it: <Ana_selects_a_dress>, <Ana_pays_for_the_dress > and <Ana_collects_her_dress>.



(a) (b)

Figure 11: Use Case refinement hidden under a decomposition: (a) Composite/Component relationship. (b) Abstraction/Refinement relationship.

Figure 12: UC inclusion at extension level.

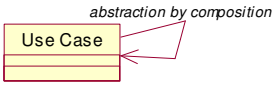


Figure 13: UC inclusion at intension level.

However, from a Use Case inclusion we can derive a model abstraction relationship called abstractions by composition (figure 13). In this case the relationship is established between models instead of being established between instances. For example, the Use Case Buy' is an abstraction of the Use Case Buy (see figure 11b. Conversely, Buy is a refined version of Buy', showing the fact of being composed by three sub Use Cases.

The mapping, reveal: Include → Abstraction, is defined to return the Abstraction artifact which can be derived from each Include artifact,

context Include **def**: reveal() : Abstraction
post: result.stereotype=<<refine>> and
result.supplier = self.base.hideParts() and
result.client = self.base
context UseCase **def**: hideParts(): UseCase -- returns an abstraction of the use case by hiding its parts--

4.2 Refining by Specialization

Use Cases are GeneralizableElements, so a Use Case may specialize a more general one. Figure 14a shows a general use case and its specialization. The general Use Case describes a Payment, while the specialization describes a particular kind of payment: PaybyCreditCard.

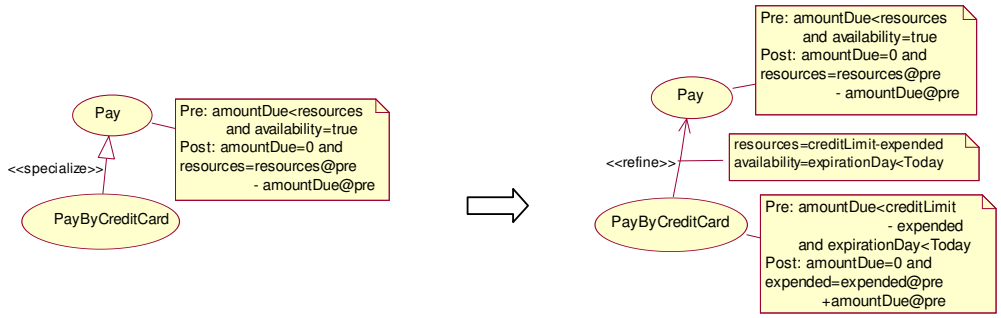
The inheritance mechanism allows modelers to describe Use Cases incrementally by describing extensions (increments) to previously defined Use Cases (its parents). Reusing in this way the more general specification.

The Figure 14 shows the connection between generalization/specialization relationship and abstraction/refinement relationship between Use Cases. The UML Generalization artifact in figure 14a relates two Use Cases: the parent, Pay, and the child, PayByCreditCard. The child is not a self contained model, it is just an increment of its parent. While, on the other hand the abstraction/refinement relationship in figure 14b relates self

contained models that are obtained by combining the parent Use Case with the child Use Case. Attributes, preconditions as well as post conditions are combined.

The mapping, reveal: Generalization → Abstraction, is defined to return the Abstraction artifact which is hidden under each Generalization artifact. The refinement is obtained by combining the parent and the child intension.

Context Generalization **def**: reveal(): Abstraction
pre: self.parent.oclIsKindOf(UseCase) and self.child.oclIsKindOf(UseCase)
post: result.stereotype=<<refine>> and
result.supplier = self.parent and result.client = (self.parent ⊕ self.child)[6] and
result.abstractionMapping(r) = $r\downarrow_{\Sigma}$ [7], where Σ= self.parent.allFeatures



(a) (b)

Figure 14: Use Case Refinement hidden under a Generalization: (a) Generalization/Specialization relationship. (b) Abstraction/Refinement relationship.

The abstraction mapping returns an abstract version of the object by keeping only the features defined in the parent Class while forgetting the rest. The implementation mapping displayed in figure 14b is not automatically produced. The designer is asked to specify the relation between the attributes of the abstract use Case (i.e. amountDue, resources and availability) and the attributes of the refined Use Case (i.e. amountDue, creditLimit, expended and expirationDate).

On the semantics side, as a consequence of the differentiation introduced by the specialization, the extension set of the abstract Use Case becomes partitioned in two or more sub sets, for example, let d be the abstraction artifact derived from the specialization of the use case Pay by the sub use cases PayByCheck and PayByCreditCard; Pay.extension is equal to
(PayByCreditCard.extension.union(PayByCheck.extension)).collect(d.abstractionMapping)

5. Tool support

The task of documenting refinement steps needs to be assisted by tools. We created PAMPERO [Pons et al., 2004] that is a plug-in to the Eclipse development environment [IBM, 2003]. It consists of four components: an UML editor, an abstraction/refinement translator, an evaluator, and a detective:

[6] Addition of Use Case intension with operation overriding is defined in the following way,
Let U = (Attr, (pre_{op}, $post_{op}$)) and U'= (Attr', ($pre_{op'}$, $post_{op'}$))
U ⊕ U' = (Attr ∪ Attr' , (pre_{op} or $pre_{op'}$, $post_{op}$ and $post_{op'}$))

[7] The restriction functor $\downarrow_{\Sigma}$ takes an Scenario and returns a restricted version of the scenario containing only those actions that are defined in Σ.

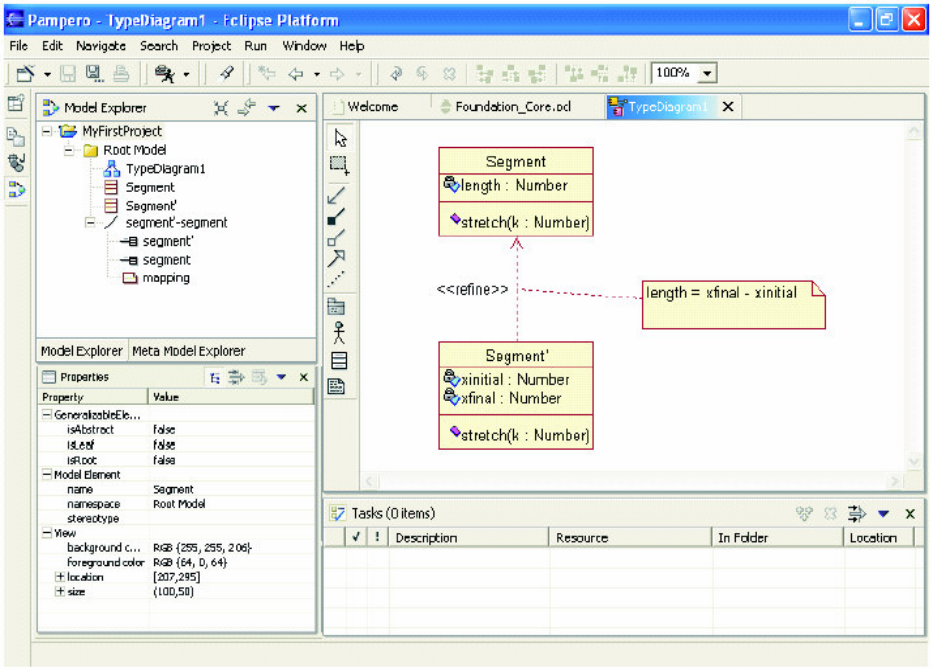


Figure 15. The PAMPERO tool: Edition of explicit refinement.

The Editor. The editor supports the creation of a number of UML artifacts, including Abstractions; see figure 15. Additionally, the editor allows developers to specify the abstraction mapping attached to Abstraction artifacts, using OCL expressions.

The abstraction/refinement Translator. The translator takes an OCL expression attached to a Class and translates it to concrete vocabularies, following the refinement steps. The translation of expressions attached to elements other than Class, is not supported yet.

The evaluator. The evaluator takes OCL expressions and evaluates them on a given model (see figure 16). Expressions might be either originally written in the model's vocabulary or translated by the translator from another abstraction level.

The Detective. This component looks into the model to discover and reveal cases of hidden refinement. The abstraction mappings automatically generated by the detective are generally in an immature state and should be completed by the developer.

Additional components, such as the *Refinement Checker*, are being currently designed to enhance the tool; the Refinement Checker will be able to prove the existence of a formal refinement relationship between model elements, for instance given an operation refinement, the checker will prove that the precondition of the abstract operation implies the precondition of its refinement and the postcondition of the abstract operation is implied by the postcondition of its refinement.

6. Conclusions

Although the UML allows for the explicit documentation of the abstraction/refinement relationship, an important amount of variations of this relationship remains unspecified, in general hidden under other notations. To enable traceability of requirements the presence of "undercover refinement" should be discovered and precisely documented.

When the mapping between the abstract and the concrete models is explicitly (and formally) documented, assertions written in the abstract model's vocabulary can be translated, following the representation mapping, in order to analyze if they hold in the implementation. Alternatively, instances of concrete models can be abstracted according to the abstraction mapping so that abstract properties can be tested on them.

The contribution of this article is to clarify the abstraction/refinement relationship in UML models, providing basis for tools supporting the refinement driven modeling process. PAMPERO, the tool reported in this article, is an

evidence of the feasibility of the proposal, although its application to industrial cases is a pending task which is essential to determine its scalability and practical advantages.

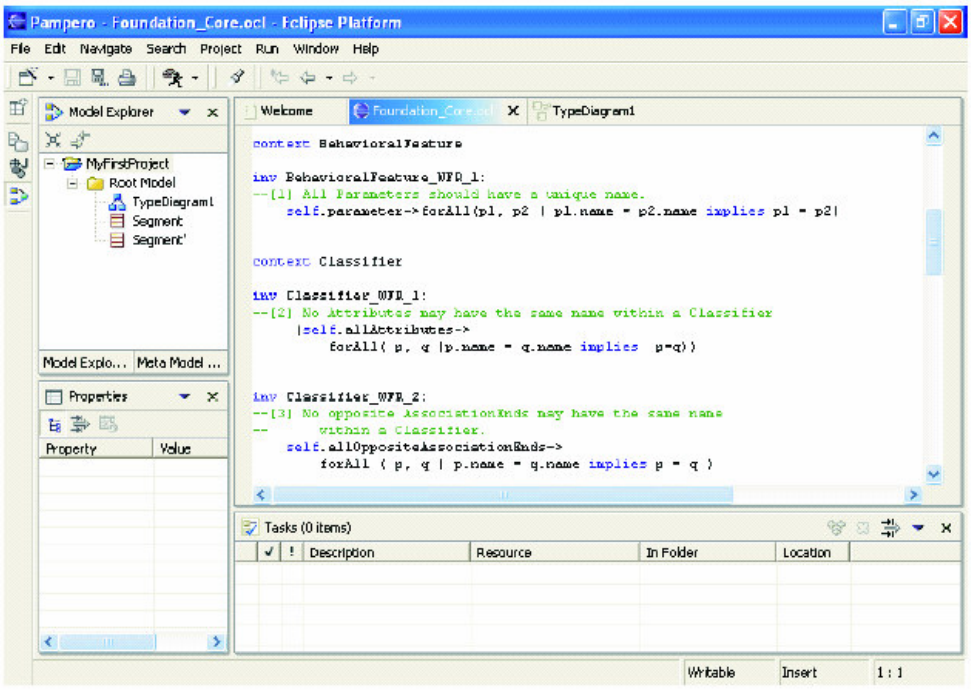


Figure 16. The PAMPERO tool: Evaluation of OCL constraints.

References

Abadi, Martin and Cardelli, Luca. A Theory of Objects, Monographs in Computer Science, Springer, 1996.

Booch, G.. Object Oriented Analysis and Design with Applications. Benjamin Cummings, 1991.

Cardelli, L.,Wegner P. On Understanding Types, Data Abstraction and Polymorphism. Computing Surveys, 17(4). 1985.

D´ Souza, Desmond and Wills, Alan. Objects, Components and Frameworks with UML.Addison-Wesley. 1998.

Dijkstra, E.W., A Discipline of Programming. Prentice-Hall, 1976.

Eric, H. and.Sernadas, A. Algebraic Implementation of Objects over Objects. In stepwise Refinement of Distributed Systems, Models, Formalism, Correctness. LNCS 430. 1989.

Giandini, R., Pons, C., Pérez,G. Use Case Refinements in the Object Oriented Software Development Process. Proceedings of CLEI 2002, ISBN 9974-7704-1-6, Uruguay. 2002.

IBM, The Eclipse Project. Home Page. Copyright IBM Corp. and others, 2000-2003. http://www.eclipse.org/.

OMG. The Unified Modeling Language Specification – Version 1.4, UML Specification, revised by the OMG, http://www.omg.org, September 2001

Pons, Claudia, Giandini Roxana and Baum Gabriel. Specifying Relationships between models through the software development process, Tenth International Workshop on Software specification and Design, San Diego., IEEE Computer Society Press. November 2000.

Pons, C., Pérez,G., Giandini, R., Kutsche, Ralf-D. Understanding Refinement and Specialization in the UML. and. 2nd International Workshop on MAnaging SPEcialization/Generalization Hierarchies (MASPEGHI). In IEEE ASE 2003, Canada.

Pons, C., Giandini, R, Pérez., G., Pesce, P., Becker, V., Longinotti,J., Cengia,J., Kutsche, R-D., The PAMPERO Project: "Formal Tool for the Evolutionary Software Development Process". Home page: http://sol.info.unlp.edu.ar/eclipse.

Steimann,F., Göβner,J, Mück,T. On the key rol of compositioning object oriented modelling. Proceedings of the 6th Int. Conference <<UML 2003>>. LNCS 2863. Springer. 2003.

Wegner, P and Zdonik, S, Inheritance as an Incremental Modification Mechanism or What like is an isn't like. in proceedings 3rd European Conference on Object-Oriented Programming (ECOOP'88), Springer, 1988.

Reengenharia de Sistemas Orientados a Objetos para Sistemas Orientados a Aspectos

Ricardo Argenton Ramos[1]
Anderson Pazin*
Rosângela Ap. D. Penteado
UFSCar - Universidade Federal de São Carlos, Departamento de Computação.
São Carlos - SP, Brasil, CEP:13565-905, 55(0xx 16) 260-8233
{ rar, rosangel }@dc.ufscar.br

*Faculdades Salesiana de Lins, Centro de Tecnologia de Informação.
Lins - SP, Brasil, CEP: 16400-505, 55(0xx14) 3522 - 4733
anderson@salesianolins.br

Abstract

The source code of non-functional concerns spread and tangled with functional concerns in Objects Oriented systems, cause problems as the agreement difficulty, reuse and to add new functionalities to the system. To bright up these problems, appears the Aspect-Oriented Programming, having as main characteristic the structures supply *that better encapsulate the concerns. This work shows the accomplishment of a process of a reengineering* process *using the Aspecting approach, where the concern of Persistence is identified, modeled and implemented in a language that supporst the Aspect-Oriented Paradigm. Two systems are used as case study, one implements the Persistence Layer design patern, and the other does not use the design patern at the persistence implementation.*

Key Words: Reengineering, Aspect Oriented Programming, Persistence.

Resumo

O código fonte de interesses não funcionais espalhados e entrelaçados com interesses funcionais em sistemas Orientados a Objetos, causa problemas como a dificuldade de entendimento, de reuso e de adicionar novas funcionalidades ao sistema. Visando amenizar problemas como esses, surge a Programação Orientada a Aspectos, que tem como principal característica o fornecimento de estruturas que melhor encapsula os interesses. Este trabalho apresenta a realização de um processo de reengenharia utilizando a abordagem Aspecting, em que o interesse de Persistência é identificado, modelado e implementado em uma linguagem que da apoio ao paradigma Orientado a Aspectos. Dois sistemas são utilizados como estudo de caso, sendo que um implementa o padrão de projeto Camada de Persistência, e o outro não utiliza o padrão na implementação da persistência.

Palavras-Chave: Reengenharia, Programação Orientada a Aspectos, Persistência.

[1] Apoio Financeiro do CNPq

1. Introdução

Sistemas legados freqüentemente possuem alto custo de manutenção, lógica desestruturada e com isso aumentam as dificuldades para a interação com novas tecnologias e novos ambientes. Porém, possuem embutidos em seu código as regras de negócio e o conhecimento de vários anos de seus desenvolvedores/mantenedores que não podem ser simplesmente descartados. Desenvolver um novo sistema utilizando novas tecnologias e conceitos, além de consumir muito esforço tem-se o receio de que todas as regras de negócio, contidas somente no código fonte possam ser perdidas.

Um sistema de software é composto por requisitos funcionais e não funcionais. Os requisitos funcionais correspondem às condições ou à capacitação que devem ser contempladas pelo software. Geralmente, trata-se de necessidades do cliente e/ou do usuário para resolver um problema ou alcançar um objetivo [5]. Os requisitos não funcionais não expressam função (transformação) a ser implementada em um sistema, mas sim condições de comportamento e restrições que devem prevalecer, como por exemplo, tratamento de exceções, persistência de dados, segurança e de desempenho [4]. Os requisitos não funcionais na maioria das vezes em sistemas Orientados a Objetos são implementados de forma entrelaçada com o código que implementa os requisitos funcionais causando problemas como a dificuldade de entendimento do código e, conseqüentemente, de reuso, de manutenção e de adição de novas funcionalidades. Uma forma de amenizar essas dificuldades é a utilização da separação dos requisitos. Os requisitos não funcionais, de agora em diante, chamados de interesses podem ser isolados e implementados com linguagens de implementação com tal capacidade.

Sistemas implementados com o paradigma Orientado a Objeto utilizam técnicas de modularização de interesses bem sucedidas, porém essa abordagem de modularização, de acordo com um único interesse inerente, é insuficiente, por não prover todas as estruturas necessárias para o desenvolvimento de sistemas complexos [17]. Embora as pesquisas em engenharia de software estejam bastante amadurecidas, a manutenção de software permanece como um problema central à área. Outras técnicas como a utilização de padrões de projeto visam minimizar o problema da modularidade, porém segundo alguns autores [12] o código que implementa o padrão de projeto ainda fica entrelaçado e espalhado pelo código funcional da aplicação, não solucionando assim o problema da dificuldade de reusar, manter o sistema.

A reengenharia de sistemas Orientados a Objetos para Orientados a Aspectos (OA), pode ser usada para minimizar as dificuldades citadas. Pois a Programação Orientada a Aspectos possui mecanismos mais eficientes para a modularização de interesses com alternativas válidas para elaborar sistemas modularizados, facilitando futuras manutenções e evoluções [10]. Tendo como motivação os problemas apresentados, Ramos [15] propõe a abordagem *Aspecting* que visa fornecer um processo para que engenheiros de software conduzam a migração de sistemas Orientados a Objetos para sistemas Orientados a Aspectos.

O objetivo deste artigo é mostrar como o interesse de persistência em banco de dados relacional existente em sistemas Orientados a Objetos é modularizado e implementado utilizando a Programação Orientada a Aspectos. Para isso, dois estudos de caso são apresentados: Um sistema que utiliza o padrão de projeto Camada de Persistência [18] para implementar o interesse de persistência e outro sistema que não utiliza padrão. A realização da reengenharia é apoiada pela abordagem *Aspecting*. Ressalta-se que essa abordagem apóia a identificação e posterior modelagem e implementação de outros interesses aqui não especificados [15].

Na Seção 2 encontram-se os assuntos relacionados a separação de interesses e a programação orientada a aspectos; na Seção 3 a abordagem Aspecting é apresentada. Na Seção 4 a aplicação da abordagem Aspecting a dois sistemas exemplo é apresentada; e as considerações finais são apresentadas na Seção 5.

2. Separação de Interesses e a Programação Orientada a Aspectos

Czarnecki e Eisenecker [6] comentam que a necessidade de manipular um requisito importante de cada vez, durante o desenvolvimento de um sistema, é chamado de princípio da separação de interesses. Linguagens de programação geralmente fornecem construtores para organizar um sistema em unidades modulares que representam os interesses funcionais da aplicação. Essas unidades são expressas como objetos, módulos e procedimentos. Mas, também, há interesses em um sistema que abrangem mais de um componente funcional, tais como: sincronização, interação de componentes, persistência e controle de segurança. Esses interesses são geralmente implementados por fragmentos de código espalhados pelos componentes funcionais. Eles são chamados de aspectos (em nível de código) e de interesses (em nível de projeto) e alguns são dependentes de um domínio específico, enquanto outros são mais gerais.

Outras técnicas que procuram auxiliar a separação de interesses é a utilização de padrões de projeto, como os propostos por Gamma e outros [8]. Porém, segundo Noda e Kishi [12], as técnicas atuais de programação não são adequadas para a utilização dos padrões de projeto por tornarem a aplicação dependente deles, diminuindo as chances de reuso da parte funcional da aplicação.

Um exemplo de padrão de projeto é o padrão Camada de Persistência que permite minimizar a incompatibilidade existente entre o paradigma orientado a objetos e a persistência de dados em um banco relacional

[18]. Ele consiste em uma camada de persistência que cuida da interface entre objetos de aplicação e tabelas de banco de dados relacional. Possui um conjunto de dez sub-padrões que podem ser utilizados para a implementação dessa camada de persistência: Camada Persistente (*Persistent Layer*), *CRUD* (Criação (**C**reate), Leitura (**R**ead), Atualização (**U**pdate) e Remoção (**D**elete)), Descrição de Código SQL (*SQL Code Description*), Gerenciador de Tabelas (*Table Manager*), Métodos de Mapeamento de Atributos (*Atributte Mapping Methods*), Conversão de Tipos (*Type Conversion*), Gerenciador de Mudanças (*Change Manager*), Gerenciador de Identificadores Únicos (*OID Manager*), Gerenciador de Conexão (*Connection Manager*) e Gerenciador de Transação (*Transaction Manager*). Cada um deles possui uma funcionalidade bem definida e enquanto alguns são obrigatórios, outros são opcionais. Por exemplo, o padrão CRUD é essencial, para a realização das operações básicas de persistência, já o padrão Conversão de Tipos é útil, mas não essencial, podendo não ser utilizado.

O padrão Camada de Persistência possui três formas de uso, segundo Cagnin [1] a terceira forma é a que permite mais facilmente reutilizar o Gerenciador de Tabelas (*Table Manager*), pois não há preocupação com a persistência de qualquer tipo de objeto no banco de dados, necessitando apenas conhecer os parâmetros dos métodos das classes. Além disso, o sistema implementado utilizando-a é o mais manutenível. Embora a classe `TableManager` seja reusável, o mesmo não ocorre com as classes da aplicação, devido à dependência do padrão, que ocorre porque essas classes invocam métodos existentes nas classes que implementam o padrão.

Com a utilização de técnicas da programação orientada a aspectos, com a intenção de melhorar a separação de interesses de sistema implementados com padrões de projeto, Camargo e outros [2] implementam com aspectos o padrão de projeto Camada de Persistência [18]. Um dos benefícios obtido com essa implementação é o reuso de código do padrão devido a inversão das dependências. O código funcional do sistema não depende do código do padrão implementado nos aspectos, como ocorre na implementação orientada a objetos. Assim, há um impacto direto na localidade de código, pois todas as dependências entre os padrões e a aplicação são localizadas no código do padrão.

A Programação Orientada a Aspectos (POA) trata os interesses que entrecortam as classes de modo análogo ao que a programação orientada a objetos faz para o encapsulamento e a herança. Ou seja, provê mecanismos de linguagem que explicitamente capturam a estrutura de entrecorte, alcançando, assim, os benefícios de melhor modularidade, tais como: código mais simples, mais fácil de manter e alterar e, conseqüentemente, aumento da reusabilidade do código [10].

A programação orientada a aspectos consiste na separação dos interesses de um sistema em unidades modulares e posterior composição (*weaving*) desses módulos em um sistema completo [6]. Os interesses podem variar de noções de alto nível, como segurança e qualidade de serviço, a noções de baixo nível, como sincronização e manipulação de *buffers* de memória.

Uma linguagem bastante difundida para o apoio a POA de propósito geral é a AspectJ [11]. Seus componentes são representados por classes Java e uma nova construção, denominada aspecto (*aspect*), é responsável por implementar os interesses que entrecortam a estrutura das classes . A composição dos aspectos com as classes é executada em tempo de compilação, sobre *bytecode* ou código fonte Java.

Os seguintes conceitos são usados em AspectJ:

Pontos de junção (*joinpoints*): são métodos bem definidos na execução do fluxo do programa que compõem pontos de corte.

Pontos de corte (*pointcuts*): identificam coleções de pontos de junção no fluxo do programa.

Sugestões (*advices*): são construções semelhantes a métodos, que definem comportamentos adicionais aos pontos de junção. São executadas quando pontos de junção são alcançados [11]. AspectJ tem três tipos diferentes de sugestões:

i) **Pré-sugestão (*before*):** é executada quando um ponto de junção é alcançado e antes da computação ser realizada.

ii) **Pós-sugestão (*after*):** é executada quando um ponto de junção é alcançado e após a computação ser realizada.

iii) **Sugestão substitutiva (*around*):** é executada quando o ponto de junção é alcançado e tem o controle explícito da computação, podendo alternar a execução com o método alcançado pelo ponto de junção.

A modularização de interesses de entrecorte é realizada usando pontos de junção (*join points*) e sugestões (*advices*).

Uma vez executada a compilação, os entrecortes estarão permanentemente incorporados às classes, não sendo possível alterar a composição durante a execução do programa. O processo de composição segue duas etapas: i) o código Java e o código AspectJ são compilados em *bytecode* Java, instrumentado com informações sobre as construções do AspectJ (como sugestões e pontos de corte); ii) o compilador utiliza essas informações para gerar as classes compostas (*bytecode* compatível com a especificação Java) [9].

3. Abordagem *Aspecting*

A abordagem é composta por três etapas distintas: I Entender a Funcionalidade do Sistema; II Tratar o Interesse e III. Comparar Sistema OA com OO, que podem ou não ser apoiadas por ferramentas CASE ou

ferramentas especificas da POA. A Figura 1 mostra as etapas (representadas por retângulos) que compreendem a abordagem *Aspecting*, utilizando a notação SADT [16].

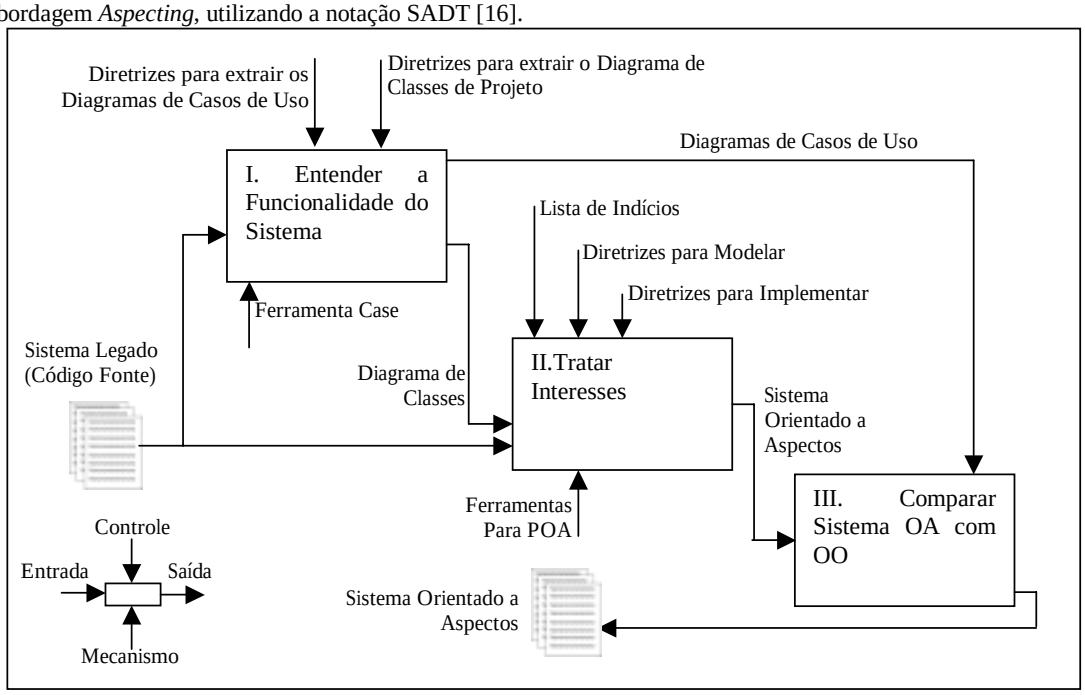


Figura 1 - Etapas da Abordagem *Aspecting*.

Cada etapa da abordagem contém diretrizes que auxiliam o engenheiro de software a conduzir a reengenharia. As etapas e os passos que compõem a abordagem são apresentados na Tabela 1.

Tabela 1 - Etapas e Passos da Abordagem *Aspecting*.

Etapas	**Passos**	**Descrição**
I. Entender a Funcionalidade do Sistema **Descrição:** Extrai as informações sobre a funcionalidade do sistema.	**I.1.** Gerar Diagramas de Casos de Uso. (caso não existam)	Executar o sistema para criar casos de uso correspondentes às funcionalidades nele existente. Utilizar as diretrizes criadas para esse fim.
	I.2. Gerar Diagrama de Classes de Projeto. (caso não exista)	Criar um diagrama de classe a partir do código fonte.Utilizar as diretrizes criadas para esse fim. **Opção**: Utilizar a Ferramenta *Omondo* [13].
II. Tratar Interesses **Descrição:** Esta etapa é evolutiva sendo os passos repetidos até que todos os interesses da Lista de Indícios sejam pesquisados ou até que o engenheiro de software decida finalizar o processo.	**II.1.** Marcar o Interesse. (caso exista)	Para um dos interesses da lista de indícios: Para cada classe implementada no código fonte, se existir tal indício Marcar o trecho do código fonte, adicionando comentários no final de cada linha, indicando o nome do interesse, seguido de um número seqüencial que deverá indicar a ordem em que esse trecho aparece na classe.
	II.2. Modelar o Interesse.	Adicionar ao diagrama de classe de projeto o aspecto. Utilizar as diretrizes específicas de cada interesse criadas para esse fim.
	II.3. Implementar o Interesse.	Implementar o aspecto que foi adicionado ao diagrama de classes no passo anterior. Utilizar as diretrizes específicas de cada interesse criadas para esse fim.Retornar ao passo 2.1.
III. Comparar Sistema Orientado a Aspectos com o Orientado a Objetos **Descrição**: Compara a funcionalidade do sistema OA que foi gerado com a do sistema OO original.	**III.1.** Comparar a Funcionalidade do Sistema Orientado a Aspectos	Utilizar os diagramas de casos de uso elaborados no passo 1.1. e suas descrições, e com o auxilio de diretrizes, criadas para esse fim, verificar se o sistema gerado atende às funcionalidades do sistema original representadas nos diagramas de casos de uso.

A identificação de interesses não funcionais espalhados pelo código fonte das classes funcionais Java é realizada buscando-se fragmentos de código, que podem ser: palavras chave, atributos, métodos, classes e objetos. Observando-se essas características no código fonte em três sistemas utilizados como estudos de caso [15],

elaborou-se a Lista de Indícios, para auxiliar o engenheiro de software a identificar os trechos de código fonte que podem conter indícios de ser de um determinado interesse. Esses indícios sinalizam a possibilidade da existência de um interesse. A Lista de Indícios é composta para auxiliar a sinalização de seis diferentes interesses, mas neste artigo somente o de persistência é utilizado. Os indícios para esse interesse são expressos na forma de expressões regulares.

Na Seção seguinte é apresentada a aplicação da abordagem Aspecting para dois sistemas implementados em Java, sendo que um utiliza o padrão de projeto Camada de Persistência [18] e o outro não utiliza.

4. Aplicação da Abordagem *Aspecting* a dois Sistemas Exemplo

Os sistemas usados neste estudo de caso são:

a) O sistema de Oficina Eletrônica efetua consertos em produtos eletrônicos como televisores, vídeo-cassetes e forno de microondas e seus técnicos recebem comissão referente aos consertos efetuados, utiliza *servlets* para a comunicação das interfaces em HTML com o banco de dados, utilizando o padrão Camada de Persistência.

b) O sistema de Caixa de Banco, obtido pela Internet, implementado na linguagem Java e que não possui documentação. Sua função é gerir contas correntes e clientes do banco.

As etapas da abordagem são realizadas seguindo os passos indicados na Tabela 1.

Etapa I - a partir da execução dos sistemas foram criados casos de uso que representasse cada funcionalidade existentes neles, de acordo com diretrizes especificas não apresentadas aqui [15]. Como exemplos de casos de uso têm-se do sistema a): cadastrar produtos, cadastrar funcionários, pagar comissão entre outros; e do sistema b) têm-se: cadastrar cliente, cadastrar contas, efetuar débitos entre outros. O diagrama de classes de cada sistema foi gerado com apoio da ferramenta *Omondo* [13]. Caso essa não seja utilizada a abordagem Aspecting fornece diretrizes para que esses diagramas sejam criados. Devido a restrição de espaço, somente será apresentada com maiores detalhes a etapa II, Tratar interesses.

Etapa II - É evolutiva, composta por três passos que são repetidos até que todos os interesses da Lista de Indícios sejam pesquisados ou até que o engenheiro de software decida finalizar o processo. Porém, como comentado anteriormente, será exibido somente o processo realizado para a identificação, modelagem e implementação do interesse de Persistência em Banco de Dados Relacional (implementado com o padrão Camada de Persistência e sem o padrão).

No **Passo II.1**: A expressão regular que auxilia o reconhecimento do interesse de Persistência em banco de Dados Relacional é mostrada na Figura 2.

<i.Persistencia.BD> = 'Connection' <statements> |'Connection' <statements> <SQL>|
<SQL> <statements> 'Connection'| 'PreparedStatement' <statements>|
'PreparedStatement' <statements> <SQL> | <SQL> <statements> 'PreparedStatement'|
'ResultSet' <statements>| 'ResultSet' <statements> <SQL> | <SQL> <statements> 'ResultSet'

Figura 2 - Indícios do interesse de Persistência em Banco de Dados Relacional.

Para o engenheiro de software conseguir identificar todo o código fonte que pertence ao interesse de Persistência no sistema de Oficina Eletrônica, foi necessário que se conhecesse a granulosidade[2] da implementação do padrão de projeto Camada de Persistência. Pois atributos pertencentes ao padrão como os mostrados na Figura 3 (a) não foram reconhecidos pela expressão regular que faz parte da Lista de Indícios, a Figura 5 (b) mostra uma linha que pode ser identificada com a expressão regular.

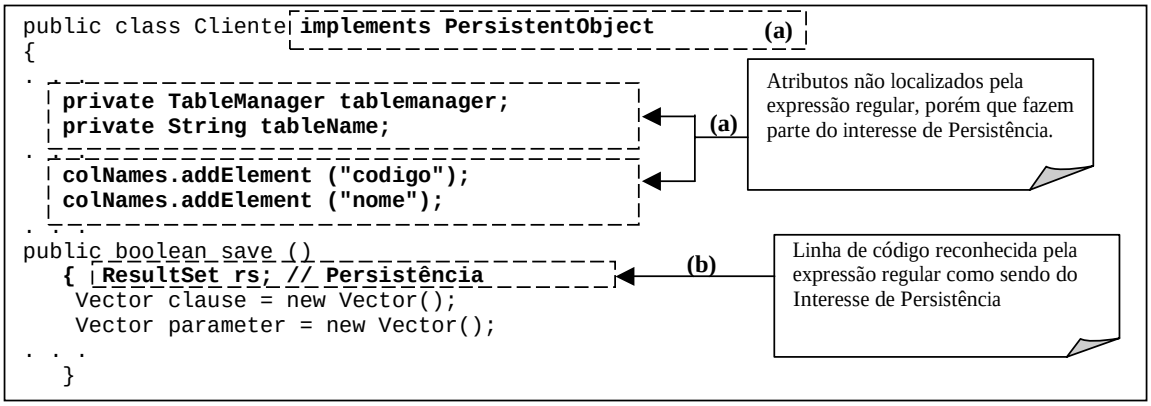


Figura 3 - Trecho de código da classe `Cliente` do sistema de Oficina Eletrônica implementado com o padrão de projeto Camada de Persistência.

[2] Como e o quanto espalhado é a implementação do padrão.

Para o sistema de Caixa de Banco a expressão regular que reconhece os indícios de persistência ajudou a identificar todos os trechos que pertencia ao interesse, como pode ser visto na Figura 4.

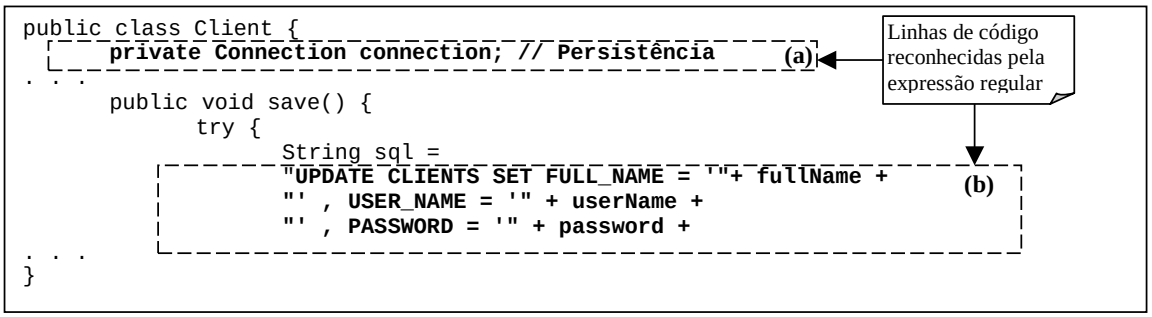


Figura 4 - Trecho de código da classe `Client` do sistema de Caixa de Banco implementado sem o padrão de projeto Camada de Persistência.

O interesse de Persistência incorpora o sub interesse de conexão com o banco de dados. Para o sistema de Oficina Eletrônica, que implementa o padrão de projeto Camada de Persistência, o sub interesse de conexão está implementado não somente nas classes que realizam a persistência, mas também nas classes *servlets* e em uma classe especifica que contém métodos para conexão e desconexão com o banco de dados. Para o sistema de Caixa de Banco, o sub interesse de conexão está somente implementado nas classes que realizam a persistência. Por esse motivo, neste artigo optou-se por não considerar a implementação do sub interesse de conexão com o banco de dados.

Passo II.2: os aspectos que modularizam o interesse de Persistência em Banco de Dados Relacional são inseridos nos diagramas de classes seguindo as diretrizes próprias do interesse exibidas na Figura 5. Essas diretrizes foram elaboradas com base em trabalhos de extensão da notação UML, para a programação orientada a aspectos, [3], [14], e na experiência obtida com a realização de estudos de caso [15].

1. Para cada classe que contenha trechos marcados como sendo do interesse de Persistência em Banco de Dados Relacional.
    1.1. Adicionar, ao diagrama de classes de projeto, um *Aspect-class* com o estereotipo <<aspect>>, seguido do Nome do aspecto.
    1.1.1. Adicionar ao *Aspect-class* os atributos do interesse, utilizando o modelo:
    [Tipo] [Nome da classe].[Atributo];
    Sendo que:
    [Tipo]: indica o tipo do atributo que foi retirado da Classe A.
    [Nome da classe]: indica o nome da Classe A.
        [Atributo]: indica-se o nome do atributo que foi retirado da Classe A.
        1.1.2. Adicionar ao *Aspect-class* os métodos que pertencem totalmente ao interesse, utilizando o modelo:
        [Tipo] [Nome da classe].[Método];
    Sendo que:
    [Tipo]: indica o tipo do retorno do método retirado da Classe A.
    [Nome da classe]: indica o nome da Classe A.
    [Método]: indica o nome do método que retirado da Classe A.
        1.1.3. Caso existam métodos que dependam de outros interesses:
            1.1.3.1 Criar um "*aspect-method*" e adicioná-lo ao *Aspect-class*, utilizando o modelo:
            (<<after>> || <<before>> || <<around>>):[Nome do Ponto de Corte]
        Sendo que:
            (<<after>> || <<before>> || <<around>>): deve-se optar pela opção dependendo da ordem em que deseja ser executar o corpo da sugestão (*advice*).
        [Nome do Ponto de Corte]: Deve-se indicar o nome do ponto de corte.
    1.2. Criar um relacionamento de associação com origem no *Aspect-class* e destino na Classe A, utilizando o estereótipo <<Introduction>>, se existirem métodos e/ou atributos relacionados ao interesse.
        1.2.1. Criar uma declaração do tipo: {atributo_1, atributo_n, método_1(), método_n()}, como papel da Classe A.
    1.3. Criar um relacionamento de associação com origem no *Aspect-class* e destino na Classe A, utilizando o estereótipo <<crosscuting>>, se existirem métodos que dependem de outros interesses.
        1.3.1. Criar uma declaração do tipo: [nome do ponto de corte], como papel do *Aspect-class*.
        Sendo que:
        [nome do ponto de corte]: indica o nome do ponto de corte.
    1.3.2.Criar um a declaração do tipo: (**?** || **#**) [ponto de junção], como papel da Classe A.
    Sendo que:
        [nome do ponto de junção]: indica a assinatura do método que contém o trecho de código marcado com interesse.

Figura 5 - Diretrizes para modelar o interesse de Persistência em Banco de Dados Relacional.

O sub padrão Gerenciador de Tabelas, presente na implementação do sistema de Oficina Eletrônica, está implementado em uma classe, `TableManager`, não tendo o espalhamento do seu código por outras classes do

sistema. Assim, optou-se por preservar a implementação existente. Na Figura 6 a classe `TableManager` contém um relacionamento de associação com os aspectos modelados, pois esses utilizam métodos dessa classe.

O diagrama de classes parcial do sistema de Oficina Eletrônica mostrado na Figura 6 apresenta os aspectos `AspectCliente` e `AspectConserto` que adicionam às classes de aplicação, métodos e atributos referentes ao padrão de projeto Camada de Persistência. Os aspectos também entrecortam as classes, interceptando seus construtores para poder atribuir valores específicos da classe aos atributos referentes ao padrão.

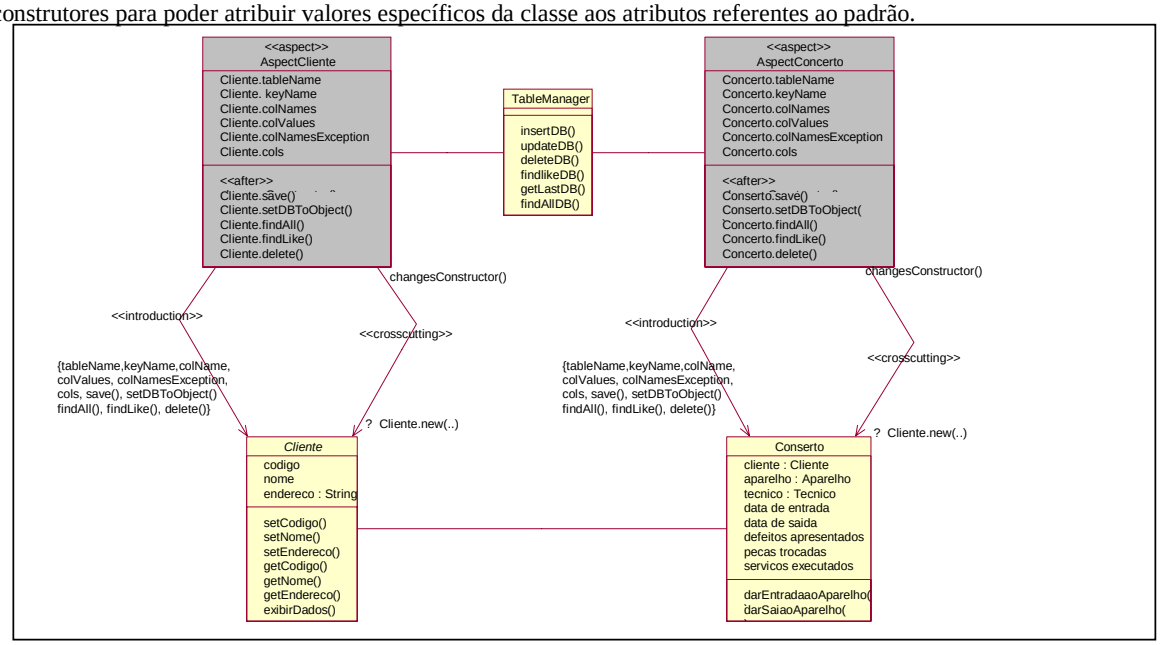


Figura 6 - Diagrama de classes parcial para o sistema de Oficina Eletrônica.

O sistema de Caixa de Banco tem em seu diagrama de classes parcial, Figura 7, dois aspectos, o `AspectClient` e `AspectAccount` que apenas introduzem métodos referentes ao interesse de persistência às classes que realizam a persistência com o banco de dados.

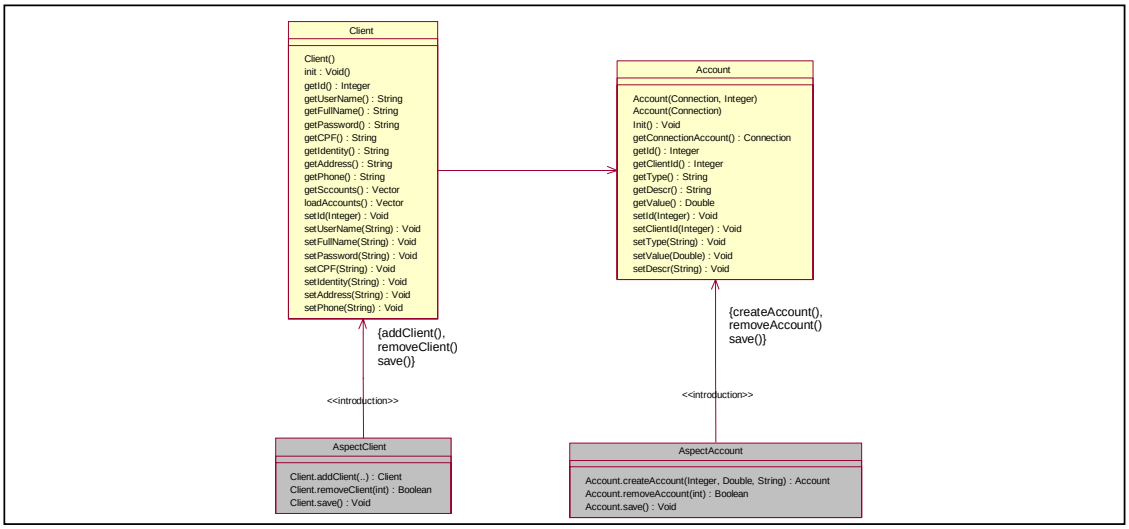


Figura 7 - Diagrama de classes parcial para o sistema de Caixa de Banco.

Os relacionamentos de associação existentes entre os aspectos e as classes rotulados com o estereótipo *<<introduction>>* indicam que no aspecto existem atributos e/ou métodos, relacionados no papel da classe, que serão introduzidos estaticamente na classe indicada, em tempo de compilação.

Já os relacionamentos de associação existentes entre os aspectos e as classes rotulados com o estereótipo *<<crosscuting>>* indicam que na classe relacionada existe um ou mais pontos de junção, relacionado no papel da classe, que compõe o ponto de corte relacionado no papel do aspecto. O sinal gráfico "?", existente no papel da classe para esses relacionamentos, Figura 6, indica a captura da execução do ponto de junção [3].

Os relacionamentos existentes não rotulados indicam a existência de uma associação entre classes e/ou entre classes e aspectos.

Passo II.3: Os aspectos que foram modelados nos dois diagrama de classes são implementados na linguagem AspectJ, seguindo as diretrizes do interesse de Persistência em Banco de Dados Relacional, Figura 8. Trechos dessas implementações para os sistemas de Oficina Eletrônica e Caixa de Banco são exibidos, respectivamente, nas Figuras 9 e 10.

Para cada *Aspect-class* criado no diagrama de classes de projeto para o interesse de Persistência em Banco de Dados Relacional.

1. Criar um aspecto em AspectJ, utilizando o modelo:
`public aspect [nome do aspecto]`
Sendo que:
`[nome do aspecto]`: é o nome indicado no *Aspect-class*.

1.1. Inserir ao aspecto os atributos, se existirem, utilizando o modelo:
`[tipo] [nome da classe].[nome do atributo];`
Sendo que:
`[tipo]`: refere-se ao tipo do atributo indicado no *Aspect-classes* do diagrama de classes de projeto.
`[nome da classe]`: refere-se ao nome da Classe A.
`[nome do atributo]`: refere-se ao nome do atributo indicado no *Aspect-classes* do diagrama de classes de projeto.

1.1.1. Utilizar comentários para marcar na classe todo o trecho de código que foi implementado.

1.2. Inserir ao aspecto os métodos que pertencem ao interesse, se existirem, utilizando o modelo:
`[tipo] [nome da classe].[nome do método] {[corpo do método]}`
Sendo que:
`[tipo]`: refere-se ao tipo do método indicado no *Aspect-classes* do diagrama de classes de projeto.
`[nome do método]`: refere-se ao nome do método indicado no *Aspect-classes* do diagrama de classes de projeto.
`[corpo do método]`: Insere-se o corpo do método da Classe A referente a esse método que esta sendo implementado no aspecto.

1.2.1. Utilizar comentários para marcar na classe todo o trecho de código que foi implementado.

1.3. Criar um ponto de corte, se existir, de acordo com as informações inseridas no diagrama de classes de projeto, utilizando o modelo:
`pointcut [nome do ponto de corte] ([objeto]): [execution || call]([ponto de junção]) && [target || this]([objeto]) && args([argumentos]);`
Sendo que `[nome do ponto de corte]`: refere-se ao nome do ponto de corte, indicado no papel do *Aspect-class* no relacionamento de entrecorte do diagrama de classes de projeto.
`([objeto])`: refere-se ao objeto que se deseja capturar.
`[execution || call]`: refere-se ao (? ‖ #) indicado no papel da Classe A no relacionamento de entrecorte do diagrama de classes de projeto.
`[target || this]`: refere-se ao indicado no papel da Classe A no relacionamento de entrecorte do diagrama de classes de projeto.
`([argumentos])`: refere-se aos argumentos indicado no papel da Classe A no relacionamento de entrecorte do diagrama de classes de projeto.

1.4. Criar uma sugestão (*advice*), utilizando o modelo:
`[after || before || around] ([argumentos]): [nome do ponto de corte] ([argumentos]) {[corpo da sugestão]}`
Sendo que: `[after || before || around]`: refere-se ao indicado no *aspect-method* do *Aspect-class* do diagrama de classes de projeto.
`[argumentos]`: são os mesmos referenciados no passo 1.3.
`[nome do ponto de corte]`: é o mesmo referenciado no passo 1.3.
`[corpo da sugestão]`: insere-se o trecho de código fonte referente ao interesse marcado no método da Classe A.

1.4.1. Utilizar comentários para marcar na classe todo o trecho de código que foi implementado

Figura 8 - Diretrizes para implementar o interesse de Persistência em Banco de Dados Relacional.

```
public aspect AspectCliente
{    public String Cliente.tableName;
     public String Cliente.keyName;
     . . .
pointcut changesConstructor(Cliente c): target (c) && execution (Cliente.new(int, String,
String, int, String, String, String, String, String, String));
      after (Cliente c): changesConstructor(c)
       {   c.setTableName("Cliente");
           c.setKeyName("codigo");
           c.setColNames("codigo");
           . . .
           id = new Integer(c.getCodigo());
           c.setColValues(id);
           c.setColValues(c.getNome());
           . . .
      public boolean Cliente.save() { . . .}
      public boolean Cliente.setDBToObject() { . . .}
      public ResultSet Cliente.findall() { . . .}
      public ResultSet Cliente.findlike() { . . .}
      public boolean Cliente.delete () { . . .}
}
```

Figura 9 - Trechos da implementação do aspecto `AspectCliente`, do sistema de Oficina Eletrônica.

```
public aspect AspectClient{
public void Client.save(){
       try {
              String sql =
              "UPDATE CLIENTS SET FULL_NAME = '"+ fullName +
              "' , USER_NAME = '" + userName +
              "' , PASSWORD = '" + password + }
              . . .
public void Client.addClient(){}

public void Client.removeClient(){}
. . .
}
```

Figura 10 - Trechos da implementação do aspecto `AspectClient`, do sistema de Caixa de Banco.

Etapa III - A comparação entre as versões dos sistemas Orientado a Objetos e dos sistemas Orientado a Aspectos foram realizada segundo as diretrizes apresentadas na Figura 11. As interfaces originais do sistema OO foram mantidas e todas as operações que anteriormente foram preservadas.

1. Se todos os casos de uso forem atendidos com sucesso, então:
    1.1. Retirar do código fonte do sistema Orientado a Aspectos, todos os trechos de código que foram marcados após a implementação dos aspectos.
2. Senão:
    2.1. Enquanto a funcionalidade do caso de uso não for atendida:
        2.1.1. Para cada interesse implementado no sistema orientado a aspecto:
            2.1.1.1.Desmarcar no código fonte os trechos desse interesse e isolar o(s) aspecto(s) implementado(s) para esse interesse.
            2.1.1.2. Verificar se a funcionalidade para o caso de uso, que não foi atendida, agora é atendida.
            2.1.1.3. Se for atendida, então:
                2.1.1.3.1. Procura-se outra forma de implementar o interesse em aspectos, ou esse interesse não é implementado.
                2.1.1.3.2. Retornar ao passo 1.1.

Figura 11 – Diretrizes para Comparar Sistema OA com OO.

5. Considerações Finais

Este artigo apresentou o processo de reengenharia realizado com a aplicação da abordagem *Aspecting* em dois sistemas Orientados a Objetos. Esses sistemas contêm o interesse de Persistência, sendo que o sistema de Oficina Eletrônica utiliza o padrão Camada de Persistência e o sistema de Caixa de Banco não utiliza. Esse interesse foi identificado nos sistemas, modelado e implementado na linguagem AspectJ. A principal contribuição deste trabalho é mostrar que o interesse de persistência pode ser implementado em aspectos utilizando a abordagem *Aspecting*, mesmo se a persistência estiver implementada seguindo o padrão de projeto Camada de Persistência.

Os sistemas resultantes deste estudo de caso possuem vantagens da boa localização de código e diminuição das redundâncias provenientes da utilização do paradigma Orientado a Aspectos, conseqüentemente o reuso, a manutenção desses sistemas é mais fácil. Porém há um aumento de módulos do sistema com a adição da estrutura aspecto (*aspect*).

Os aspectos implementados para o interesse de Persistência podem ser refinados a fim de se tornarem abstratos o bastante para obter maiores vantagens quanto ao reuso. A elaboração de um *Framework* Orientado a Aspectos que está sendo realizada atualmente pode ser talvez apontado como uma solução mais eficaz para o reuso da implementação do interesse de Persistência.

A expressão regular que faz parte da abordagem *Aspecting* não foi eficaz para encontrar todos os indícios do interesse de Persistência quando foi aplicada na versão implementada com o padrão. Porém, na versão onde o interesse de Persistência não utiliza um padrão a lista auxiliou a encontrar todos os indícios do interesse. Esse fato ocorreu, pois a expressão regular foi elaborada a partir de um sistema que não continha o padrão Camada de Persistência. Os novos indícios encontrados na implementação do Padrão Camada de Persistência foram adicionados na expressão regular, para que em novos estudos de caso essa expressão regular seja mais eficiente.

No processo realizado, deseja-se reusar o código fonte existente, separando o interesse que estava espalhado e entrelaçado. Porém se o processo for de engenharia avante o interesse de Persistência pode ser considerado isoladamente. Desse modo, as diretrizes mostradas neste trabalho para modelagem do interesse e implementação em aspectos, podem ser seguidas, sem que a marcação do interesse no código fonte original tenha de ocorrer.

6. Referências

[1] Cagnin, M.I. Avaliação das Vantagens quanto à Facilidade de Manutenção e Expansão de Sistemas Legados Sujeitos à Engenharia Reversa e Segmentação. Dissertação de Mestrado, DC-UFSCar, 1999.

[2] Camargo, V.V.; Ramos, R.A.; Penteado, R.A.D.; Masiero, P.C. Projeto Baseado em Aspectos do Padrão Camada de Persistência. In: Simpósio Brasileiro de Engenharia de Software (SBES), Manaus, 2003.

[3] Camargo, V.V. Masiero, P.C. UML-AOF – Um Perfil UML para o Projeto de Frameworks Orientados a Aspectos. Relatório Técnico, ICMC-USP, 2004. Disponível para download em: *http://www.icmc.usp.br/~valter/publicacoes*.

[4] Chung, L.; Nixon, B.; Yu, E.; Mylopoulos, J. *Non-functional requirements in software engineering*. In: *Boston: Kluwer Academic*, pág. 439, 1999.

[5] Cysneiros, L.M.; Leite, J.C.S.P. Definindo Requisitos Não Funcionais. In: Simpósio Brasileiro de Engenharia de Software (SBES´97), pág. 49-54, Outubro 1997.

[6] Czarnecki, K.; Eisenecker, U. *Generative Programming: Methods, Tools, and Applications*. Addison-Wesley, 2000.

[7] Elrad, T. Aksit, M.; Kiczales, G.; Lieberherr, K.; Ossher, H.. *Discussing Aspects of AOP*. In: Anais do ACM, pág. 33 – 38, 2001.

[8] Gama, E., Helm, R., Johnson, R., Vlissides, J. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1995.

[9] Hilsdale, E.; Hugunin, J. *Advice Weaving in AspectJ*. Submetido à *3rd International Conference on Aspect-Oriented Software Development* – AOSD. abril 2004.

[10] Kiczales, G.; Lamping, J.; Mendhekar, A. *RG: A Case-Study for Aspect-Oriented Programming*. Artigo Técnico, *SPL97*. Xerox Palo Alto Research Center, 1997.

[11] Kiczales, G.; Hilsdale, E.; Hugunin, J.; Kersten, M.; Palm, J. *Griswold, W.G. Getting Started with AspectJ*. In: Anais do ACM, pág. 59-65, Outubro 2001.

[12] Noda, N. and Kishi, T. "*Implementing Design Patterns Using Advanced Separation of Concerns*", in *Proceedings of OOPSLA 2001, Workshop on Advanced Separation of Concerns in Object-Oriented Systems*, Tampa Bay, FL, October 2001.

[13] Omondo – Ferramenta para Modelagem – *Plug-in* disponível em http://www.omondo.com.Último acesso em 04/2004.

[14] Pawlak, R., Duchien, L., Florin G., Legong-Aubry, F., Seinturier, L, Martelli, L. *A UML Notation for Aspect-Oriented Software Design*. In: *Workshop of Aspect Oriented Modeling with UML of Proceedings of Aspect Oriented Software Development Conference (AOSD) 2002*, Enschede, Abril, 2002.

[15] Ramos, R., A. *Aspecting*: Abordagem para Migração de Sistemas OO para Sistemas AO. Dissertação de Mestrado. Programa de Pós Graduação em Ciência da Computação, Universidade Federal de São Carlos, São Carlos - SP, maio de 2004.

[16] Ross, D., T. *Structure Analysis (SA): A language for communicating Ideas*. In: *IEEE Transaction Software Engineering*, 1997.

[17] Tarr, P.; Ossher, H.; Harrision, W.; Sutton, S. M. *N Degrees of Separation: Multi-Dimensional Separation of Concerns*. In: *International Conference on Object-Oriented Programming (ICSE)*, 1999.

[18] Yoder, J. W.; Johnson, R. E.; Wilson, Q. D. Connecting Business Objects to Relational Databases. Conference on the Pattern Languages of Programs, 1998.

El Problema de la Asignación de Evaluadores para los Artículos Presentados a un Evento Académico: Modelamiento e Implementación de dos Soluciones Usando Programación por Restricciones

Jesús Alexander Aranda B, Juan Francisco Díaz F, James Jerson Ortiz V
Universidad Del Valle, Escuela de Ingeniería de Sistemas y Computación
Cali, Colombia
{jesarana,jdiaz,jaortiz}@eisc.univalle.edu.co

Abstract

In this paper, we present a model of the combinatorial problem that we call *The Article Evaluators Assignment in an Academic Event.* We present two different models for it. We implement these models in a Concurrent Constraint Programming called MOzArt. The distribution strategies used for each model are equivalent; then the results are model dependent. Finally the analysis of models is based on the experimental results, and it concludes that with both models competitive results are obtained, as much in time like in quality, when approaching problems of small and medium size. Nevertheless, when dealing with problems great dimensions one of the models surpass clearly to the other, as much in quality of the solution like in efficiency, because its space search is smaller.

Keywords: Constraint Programming, Constraint Satisfaction Problems, Combinatorial Optimization Problems.

Resumen

En este artículo, se presenta el modelaje de un problema combinatorio, *El Problema de la Asignación de Evaluadores para los Artículos Presentados a un Evento Académico.* El modelaje se realiza de dos formas diferentes; posteriormente se utilizan estos modelos para realizar dos implementaciones en un lenguaje que incorpora el paradigma de programación por restricciones (MOzArt). La estrategia de distribución usada es la misma en ambos modelos, con el fín de que los resultados obtenidos sean dependientes del modelo. Por último se realiza un análisis de los dos modelos con base en los resultados experimentales, y se concluye que con ambos modelos se obtienen resultados competitivos, en terminos de tiempo y calidad, al abordar problemas de tamaño pequeño y mediano. Sin embargo, al tratar problemas de grandes dimensiones uno de los modelos supera claramente al otro, tanto en calidad de la solución como en eficiencia, debido a que su espacio de búsqueda es más pequeño.

Palabras claves: Programación por Restricciones, Problemas de Satisfacción con Restricciones, Optimización de Problemas Combinatorios.

1. INTRODUCCIÓN

Muchos problemas de la vida real pueden ser modelados como problemas de satisfacción de restricciones (**CSPs**[1]) [5]. Para cada problema se pueden desarrollar diferentes modelos; cada uno tiene su propia representación, su propia estrategia de solución y su manera particular de representar las restricciones. *El Problema de la Asignación de Evaluadores para los Artículos Presentados a un Evento Académico*, el cual clasifica precisamente en esta categoría, es el objeto de estudio de este artículo.

Un evento o congreso académico se centra en una serie de conferencias, en las cuales se exponen diferentes trabajos o artículos de investigación, los cuales son previamente referenciados y seleccionados por un Comité de Programa. Estos artículos pueden ser de diferentes áreas de investigación pero relacionadas con el tema central del congreso. Cada representante en el comité de programa, a su vez, tiene un grupo de trabajo que le colabora en el proceso de evaluación de esos artículos.

[1] del Inglés **Constraints Satisfaction Problems**

El comité de programa recibe los artículos, y de acuerdo a ciertos criterios, que toman en cuenta las fortalezas de los evaluadores y los temas de los artículos, entre otros, los asigna para evaluación de manera que cada artículo sea evaluado por un número dado de evaluadores que el evento define (3 es el número por defecto).

Dos aspectos críticos del proceso son:

1. La calidad de la solución (entendida como la adecuación de ésta a todos los criterios de distribución).
2. El tiempo que toma calcular la solución.

El primer aspecto busca minimizar la tarea de reasignación de evaluadores a causa de la incorformidad de éstos con lo que se les asignó,y asegura la pertinencia y adecuación del artículo asignado y sus respectivos evaluadores. El segundo aspecto busca facilitar la tarea del comité de programa en cuanto a que de él depende que se puedan probar diferentes soluciones y escoger la de mejor calidad entre ellas. Para el caso de la organización de la Conferencia Latinamericana de Informática del **CLEI** [2], este proceso puede tomar aproximadamente tres días y la solución encontrada genera un buen número de insatisfacciones entre los evaluadores.

Es interesante entonces, contar con una aplicación que permita encontrar lo más rápidamente posible una solución que maximice su calidad (o sea, minimice el número de inconsistencias entre los criterios de asignación y la solución real).

Con base en lo anterior, se planteó el diseño y construcción de tal aplicación. Los aspectos relacionados con los modelos utilizados y su implementación usando programación por restricciones se presentan en este artículo.

Trabajos como los presentados en [1], [2], [4] y [7] usan la programación por restricciones en la solución de problemas combinatorios. El aporte de nuestro trabajo radica en: el planteamiento de dos modelos usando restricciones para el problema presentado aquí, una implementación que ofrece resultados para problemas de dimensiones reales y la comparación entre los dos modelos usando el mismo paradigma de programación.

Inicialmente, se realiza una presentación del Paradigma de Programación por Restricciones en la sección 2 y una descripción detallada del problema en la sección 3. Posteriormente, se presentan dos modelos del problema (ver sección 4), algunos detalles de su implementación (ver sección 5), y un análisis de los dos modelos (ver sección 6). Para finalizar se presentan las conclusiones de este trabajo (ver sección 7).

2. EL PARADIGMA DE PROGRAMACIÓN CONCURRENTE POR RESTRICCIONES (CCP)

Este modelo computacional es un elegante sucesor de la programación lógica (*a la prolog*) [9]. Ambos modelos están basados en la analogía *programa = formula, solución = demostración* según la cual el programador escribe una especificación lógica del problema (definiendo variables y restricciones entre éstas) a la cual se le puede aplicar un procedimiento formal de demostración o búsqueda que eventualmente encuentra las soluciones a un problema.

Resolver un CSP con programación concurrente por restricciones consiste entonces en un ejercicio fundamentalmente de dos pasos: uno de modelación del problema (correspondiente a la especificación lógica del problema) y otro de especificación de la estrategia de búsqueda de soluciones al problema.

La modelación consiste básicamente en escribir en el lenguaje de programación por restricciones cuales son las variables, los dominios de éstas y las restricciones entre ellas.

Por cada restricción, un lenguaje de programación por restricciones crea un *propagador,* un proceso computacional que intenta *reducir el dominio* de las variables asociadas a él (es decir, descarta valores, de los dominios de las variables, que no pueden ser parte de ninguna solución).

Este conjunto de propagadores actuando sobre el conjunto de variables (reduciendo sus dominios) puede llegar a una contradicción o a un *estado de punto fijo,* un estado en el cual no se puede deducir nada nuevo acerca de las variables (es decir, no se pueden reducir más los dominios). En el primer caso se dice que a partir de ese estado no hay solución posible. En el segundo caso, se hace necesaria una etapa de búsqueda.

La especificación de la estrategia de búsqueda consiste en definir los criterios para orientar la búsqueda cuando la propagación ha llevado la computación a un estado de punto fijo en el cual existen variables no determinadas. La idea fundamental es añadir al estado de punto fijo actual, una o más restricciones que permitan que el proceso de propagación avance un poco más (es decir, que reduzca algún dominio de alguna variable). Como esas restricciones no hacen parte de las restricciones del problema, es necesario explorar también qué pasa si se añaden las restricciones contrarias al estado de punto fijo. De esta manera se exploran

[2] Centro Latinoamericano de Estudios en Informática, http://www.clei.cl

todas las posibilidades y se garantiza que si hay solución, se encuentra. Al proceso de definición de las restricciones a agregar se le suele llamar *distribución*.

¿Qué restricciones añadir entonces? Eso es lo que se especifica en la estrategia de búsqueda. Sean cuales sean, el añadirlas crea dos nuevos estados (de búsqueda); en cada uno de ellos se vuelve a aplicar el proceso de propagación hasta alcanzar un nuevo estado de punto fijo y repetir este procedimiento. Si todas las variables quedan determinadas, se ha encontrado una solución. Si después de explorar todas las posibilidades todos los estados son de contradicción, entonces el problema no tiene solución.

La eficiencia de esta búsqueda está directamente ligada al número de estados explorados. Entre más se reduzcan los dominios de las variables en la etapa de propagación, menos estados se generan y por tanto el problema se resuelve más rápidamente. El número de estados generados también depende de la estrategia de distribución. La imposición de restricciones redundantes y de restricciones que eliminen soluciones simétricas es un mecanismo bastante utilizado para aumentar la eficiencia de la búsqueda.

En conclusión, el arte de programar con restricciones consiste en:

- Encontrar una representación lógica del problema mediante restricciones, en el lenguaje de programación con que se cuenta.
- Definir una estrategia de búsqueda adecuada a la "topología del problema" que sea computacionalmente eficiente.

Hay dos clases de problemas para los que se estima conveniente utilizar programación por restricciones:

- Problemas de satisfacción de restricciones.
- Problemas de optimización combinatoria.

En los problemas de satisfacción de restricciones (**CSP**) se busca obtener soluciones factibles. En los problemas de optimización combinatoria se busca, además, que la solución factible conduzca a la minimización (o maximización) de una función objetivo. La estructura particular de estos dos tipos de problemas, se adapta a la metodología empleada por **CCP** para resolverlos.

Una representación de un **CSP** se caracteriza por lo siguiente:

- Variables:
  - X_1,X_2,...,X_n
- Valores posibles (dominios):
  - D_1,D_2,...,D_n

  Donde D_j es el dominio de X_j. El dominio puede ser finito o infinito.

- Restricciones:
  - $f(X_1,\ldots,X_n) \in [0,1]$

  Las restricciones son funciones (predicados, realmente).

Un dominio puede ser cualquier conjunto. Por ejemplo los enteros $1 \ldots 100$ o los nombres {Pedro, Juan, María} o los reales en el intervalo [0,50]. Similarmente, las restricciones pueden ser de muchas clases. Por ejemplo:

- Restricciones lógicas: Si cierta variable cumple con alguna propiedad, por ejemplo, Caudal $>$ 67.5 entonces otra variable también, por ejemplo, Apertura= 0.
- Restricciones globales: Todos los valores de $X_1,\ldots,X_n$ deben ser diferentes.
- Meta restricciones: El número de veces que 5 aparece en el arreglo A es exactamente igual a 3.
- Restricciones de elemento: El costo de asignar a la persona i la tarea j es costo[tarea[i]], cuando tarea[i]=j.

En este contexto, la programación concurrente por restricciones dispone de:

- Una metodología de modelamiento para identificar convenientemente las variables, las restricciones y la función objetivo (si la hay) de un problema.
- Un lenguaje de programación **MOzArt** [3], [8], [9], para expresar algoritmos de búsqueda que encuentren valores de las variables que satisfagan todas las restricciones y optimicen el objetivo.
- Un sistema de programación que incluye: Restricciones predefinidas con algoritmos poderosos de filtrado o propagación de valores, que reduzcan de manera efectiva el tamaño del espacio de búsqueda. Funcionalidad para la definición de nuevas restricciones y algoritmos de filtrado.

El grupo de investigación **AVISPA**[3] cuenta entre sus trabajos con aplicaciones desarrolladas bajo este paradigma, tales como **PATHOS** [7], un sistema de apoyo al planeamiento de horarios de clase aplicado a universidades y **VISiR** [2], un software de soporte para la toma de decisiones de vertimiento de agua en la represa del Alto Anchicayá.

3. DESCRIPCIÓN DEL PROBLEMA

De acuerdo a los organizadores del **CLEI**, el tiempo que toma el proceso de distribución de artículos es de 3 a 4 días. La mayor dificultad radica en poder asignar evaluadores suficientes a cada uno de los artículos cumpliendo con ciertas restricciones que se tienen en el evento (restricciones que serán descritas posteriormente). Adicionalmente, la distribución resultante presenta no pocas asignaciones de artículos que no cumplen con las restricciones.

Considerando lo anterior, la organización del evento **CLEI** 2005 que se va a realizar en Colombia en el 2005, ha desarrollado una herramienta que brinde soporte al proceso de distribución de artículos, buscando con ello, reducir el tiempo que demora el proceso y minimizar el número de asignaciones que no cumplen con todas las restricciones planteadas para el evento.

Los elementos principales en el proceso de distribución de artículos en el evento académico del **CLEI** son los siguientes:

- **Un conjunto de artículos o trabajos:** La cantidad de artículos enviados a un evento como **CLEI** 2005 se estima en aproximadamente 300 y a partir de este conjunto se seleccionan aquellos que se presentarán en las conferencias.
- **Un grupo de evaluadores:** Un artículo es revisado por 3 evaluadores; por lo tanto,si llegan 300 artículos al evento, se necesitarían 900 evaluaciones las cuales serían realizadas por los evaluadores que conforman el comité de programa.
- **Un grupo de evaluadores:** Conforman el comité del programa. Para un evento como el **CLEI** 2005 suele tener 80 miembros aproximadamente.
- **Un conjunto de restricciones:** Son las condiciones que deberían cumplirse en cualquier distribución de los artículos o trabajos recibidos.

Las restricciones que se deben tener en cuenta al momento del proceso de distribución de los artículos son las siguientes:

- **Restricción 1** :El número de evaluaciones por artículo debe ser superior o igual al mínimo requerido.
- **Restricción 2**: El número de evaluaciones por artículo debe ser inferior o igual al máximo requerido.
- **Restricción 3**: El número de artículos asignados a cada evaluador debe ser inferior o igual a su capacidad.
- **Restricción 4**: Para cada artículo, cada una de las evaluaciones las debe realizar un evaluador diferente.
- **Restricción 5**: El país del artículo debe ser diferente al país de sus evaluadores.
- **Restricción 6**: Alguno de los temas principales del artículo debe coincidir con alguno de los temas que domina el evaluador asignado.

[3] **A**mbientes **VIS**uales de **P**rogramación **A**plicativa: grupo de investigación conformado por profesores y estudiantes de las Universidades del Valle y Javeriana de Cali. Se encuentra actualmente llevando a cabo investigaciones que giran alrededor del paradigma **CCP**. Los integrantes de este grupo están actualmente desarrollando CREAR.

- **Restricción 7**: El idioma en el que está escrito el artículo, debe coincidir con alguno de los idiomas que domina el evaluador asignado.

Durante el proceso de distribución, se busca minimizar el número de asignaciones que no cumplan las anteriores restricciones. Cuando no es posible que un artículo sea asignado de tal manera que cumpla con todas las preferencias, se asigna considerando cuales restricciones son más importantes, de tal manera que algunas se cumplan y otras no. La organización del evento también puede considerar que ciertas preferencias son de obligatorio cumplimiento y por ende puede ocurrir que no se logre una distribución total. En tal caso, las asignaciones faltantes son analizadas por la organización del evento con miras a encontrar una solución.

La herramienta desarrollada por nosotros, considera los diversos factores expuestos brindando una solución al proceso de distribución.

4. MODELOS

A continuación se presentan dos modelos para el problema; en ambos se consideran los siguientes parámetros:

- **Parámetros**
  - m : Número de evaluadores.
  - n : Número de artículos.
  - $nTemas$: Número de temas del evento.
  - $minEvaArt$: Número mínimo de evaluadores por cada artículo.
  - $maxEvaArt$: Número Máximo de evaluadores por cada artículo
  - $paisArticulo_i$: El país del artículo i. $\forall i = 1, \ldots, n$.
  - $temasArticulo_i$: El conjunto de los temas del artículo i. $\forall i = 1, \ldots, n$.
  - $idiomaArticulo_i$: El idioma en que está escrito el artículo i. $\forall i = 1, \ldots, n$.
  - $paisEvaluador_j$: El país del evaluador j. $\forall j = 1, \ldots, m$.
  - $temasEvaluador_j$: El conjunto de temas del evaluador j. $\forall j = 1, \ldots, m$.
  - $idiomasEvaluador_j$: El conjunto de los idiomas en que puede evaluar el evaluador j. $\forall j = 1, \ldots, m$.
  - $capacidadEvaluador_j$: El número de evaluaciónes que puede realizar el evaluador j. $\forall j = 1, \ldots, m$.

4.1. Modelo Binario

- **Variables de Decisión**
  - $dom_{i,j} = \begin{cases} 1 & Si \ el \ artículo \ i \ es \ asignado \ al \ evaluador \ j; \ \forall i = 1, ..., n, \forall j = 1, ..., m \\ 0 & sino \end{cases}$
  - cp_i: Número de evaluadores asignados al artículo i; $\forall i = 1, ..., n$.
  - $c = \sum_{i=1}^{n} cp_i$: Número total de evaluaciones asignadas, donde $n * minEvaArt \leq c \leq n * maxEvaArt$.
- **Función Objetivo**
  - $Maximizar \ c = \sum_{i=1}^{n} cp_i = \sum_{i=1}^{n} \sum_{j=1}^{m} dom_{i,j}$
- **Restricciones**
  - Restricción 1:

$$minEvaArt \leq cp_i = \sum_{j=1}^{m} dom_{i,j}, \forall i = 1, \ldots, n$$

- Restricción 2:

$$cp_i = \sum_{j=1}^{m} dom_{i,j} \leq maxEvaArt, \forall i = 1, \ldots, n$$

- Restricción 3:

$$capacidadEvaluador_j \geq \sum_{i=1}^{n} dom_{i,j}, \forall j = 1, \ldots, m$$

- Restricción 4: Se garantiza por el modelo.
- Restricción 5:

$$paisArticulo_i \neq paisEvaluador_j, \quad \forall i = 1, \ldots, n, \quad \forall j \in \{1, \ldots, m\} : dom_{i,j} = 1$$

- Restricción 6:

$$temasArticulo_i \cap temasEvaluador_j \neq \phi, \quad \forall i = 1, \ldots, n, \quad \forall j \in \{1, \ldots, m\} : dom_{i,j} = 1$$

- Restricción 7:

$$idiomaArticulo_i \in idiomasEvaluador_j, \quad \forall i = 1, \ldots, n, \quad \forall j \in \{1, \ldots, m\} : dom_{i,j} = 1$$

4.2. Modelo Alternativo

- **Variables de Decisión**
  - $dom_{i,k} = \begin{cases} j & Si \ la \ k-ésima \ evaluación \ del \ artículo \ i \ es \ asignada \ al \ evaluador \ j, \\ & \quad para \ \ i \in \{1, \ldots, n\}, \ \ j \in \{1, \ldots, m\} \ \ y \ \ k \in \{1, \ldots, maxEvaArt\} \\ 0 & si \ no \ se \ pudo \ asignar \ ningún \ evaluador. \end{cases}$
  - c : Número total de evaluaciones asignadas, donde $n * minEvaArt \leq c \leq n * maxEvaArt$.

- **Función Objetivo**
  - $Maximizar \ c = |\{(i,k) : dom_{i,k} \neq 0, 1 \leq i \leq n, 1 \leq k \leq maxEvaArt\}|$
- **Restricciones**
  - Restricción 1:

$$minEvaArt \leq |\{(k,j); dom_{i,k} = j, 1 \leq j \leq m, 1 \leq k \leq maxEvaArt\}|, \quad \forall i = 1, \ldots, n$$

  - Restricción 2:

$$maxEvaArt \geq |\{(k,j); dom_{i,k} = j, 1 \leq j \leq m, 1 \leq k \leq maxEvaArt\}|, \quad \forall i = 1, \ldots, n$$

  - Restricción 3:

$$capacidadEvaluador_j \geq |\{(i,k); dom_{i,k} = j, 1 \leq i \leq n, 1 \leq k \leq maxEvaArt\}|, \quad \forall j = 1, \ldots, m$$

  - Restricción 4:

$dom_{i,j} \neq dom_{i,k}$, $Donde \ j \neq k, \forall i = 1, \ldots, n, \ \forall j = 1, \ldots, maxEvaArt, \ \forall k = j{+}1, \ldots, maxEvaArt$

  - Restricción 5:

$$paisArticulo_i \neq paisEvaluador_j, \quad Si \ \ \forall i = 1, \ldots, n \ \ y \ \ \forall j = 1, \ldots, m$$
$$y \ \ \exists k \in \{1, \ldots, maxEvaArt\} : dom_{i,k} = j$$

  - Restricción 6:

$$temasArticulo_i \cap temasEvaluador_j \neq \phi, \quad Si \ \ \forall i = 1, \ldots, n, \ \ \forall j = 1, \ldots, m$$
$$y \ \ \exists k \in \{1, \ldots, maxEvaArt\} : dom_{i,k} = j$$

  - Restricción 7:

$$idiomaArticulo_i \in idiomasEvaluador_j, \quad Si \ \ \forall i = 1, \ldots, n, \ \ \forall j = 1, \ldots, m$$
$$y \ \ \exists k \in \{1, \ldots, maxEvaArt\} : dom_{i,k} = j$$

5. IMPLEMENTACIÓN USANDO PROGRAMACIÓN POR RESTRICCIONES

En esta sección se presentan algunos aspectos de la implementación de los dos modelos usando programación por restricciones mediante el lenguaje de programación **MOzArt**.

Una de las ventajas de la programación por restricciones es la naturalidad con que se puede transcribir un modelo matemático en un lenguaje de programación con dicho paradigma. El propósito de esta sección es mostrar dicha naturalidad.

Por restricciones de espacio, no se presenta la implementación de todas las restricciones del problema. Sin embargo, se concibieron con la misma naturalidad de las aquí presentadas.

5.1. Variables

El dominio de las variables se definió y se presenta en los cuadros 1 y 2.

Variable	Descripción	Implementación
C	Número total de asignaciones	$\{FD.int \;\; NumArticulos * MinEvaArt \# NumArticulos * MaxEvaArt \;\; C\}$
Dom	Tupla con las asignaciones	$\{FD.tuple \;\; articulos \;\; NumArticulos * NumEvaluadores \; 0\#1 \;\; Dom\}$

Cuadro 1: Variables del Modelo Binario

Variable	Descripción	Implementación
C	Número total de asignaciones	$\{FD.int \;\; 0\#(NEA * NumArticulos - 1) \;\; C\}$
Dom	Tupla con las asignaciones	$\{FD.tuple \;\; articulos \;\; NEA * NumArticulos \; 0\#NumEvaluadores \;\; Dom\}$

Cuadro 2: Variables del Modelo Alternativo

5.2. Propagadores

A continuación se presentan los propagadores para algunas de las restricciones planteadas en los modelos. En la programación por restricciones, las restricciones de un modelo se expresan utilizando propagadores.

Los **propagadores** son agentes computacionales concurrentes que se encargan de reducir el dominio de las variables que hacen parte de la restricción, eliminando valores, que no cumplen con la restricción especificada.

5.2.1. Modelo Binario

- Restricción 3:

```
{For 1 NumEvaluadores 1
       proc {$ I}
         {FD.sum
            {Map ListaArticulos fun{$ J}
                                  Dom.((I-1)*NumArticulos + J)
                                end
            }
           '=<:'capacidadEvaluador.I
         }
       end
}
```

En el modelo binario, Dom.((I-1)*NumArticulos + J) corresponde a $dom_{i,j}$ (variables de decisión) y ListaArtículos corresponde a una lista cuyos elementos son: 1,2,3, ..., NumArticulos.

En el código anterior, se están contando las asignaciones de artículos que corresponden al evaluador J, y se está imponiendo que el número de asignaciones que le corresponden al evaluador debe ser inferior o igual a su capacidad máxima, lo anterior se realiza utilizando el propagador **FD.sum**. El anterior proceso se realiza para cada uno de los evaluadores, utilizando un ciclo **FOR**[4].

- Restricción 7:

```
{For 1 NumArticulos 1
       proc {$ I}
         {For 1 NumEvaluadores 1
                proc {$ J}
                 if {Bool.'not' {List.member IdiomaArticulo.I IdiomaEvaluadores.J}}
                 then
                    Dom.((J-1)*NumArticulos + I)=:0
                 end
                end
         }
       end
}
```

Por medio de la instrucción **if**, que aparece en el código anterior, se está determinando si el idioma del artículo I no corresponde a los idiomas que maneja el evaluador J; de ser así, se impone que al artículo I no se le puede asignar el evaluador J (esto se indica con el número 0). El anterior proceso se realiza para cada uno de los evaluadores y para cada uno de los artículos, utilizando dos ciclos anidados **FOR**.

- Restricción 5:

```
{For 1 NumArticulos 1
       proc {$ I}
         {For 1 NumEvaluadores 1
                proc {$ J}
                  if PaisArticulo.I == PaisEvaluador.J
                  then
                     Dom.((J-1)*NumArticulos + I)=:0
                  end
                end
         }
       end
}
```

Por medio de la instrucción **if**, que aparece en el código anterior, se está determinando si el país del artículo I es el mismo país que el del evaluador J; de ser así, se impone que al artículo I no se le puede asignar el evaluador J (esto se indica con el número 0). El anterior proceso se realiza para cada uno de los evaluadores y para cada uno de los artículos, utilizando dos ciclos anidados **FOR**.

5.2.2. Modelo Alternativo

- Restricción 3:

```
{For 1 NumEvaluadores 1
            proc {$ I}
               {FD.atMost  capacidadEvaluadores.I  Dom I}
            end
}
```

En este modelo Dom corresponde a una lista con todas las variables de decisión $dom_{i,k}$.

En el código anterior, se está imponiendo que el evaluador I esté asignado (en Dom) un número de veces igual o inferior a su capacidad, esto se logra utilizando el propagador **FD.atMost**. Lo anterior se realiza para cada uno de los evaluadores utilizando un ciclo **FOR**.

[4] {For I1 I2 I3 P} aplica el procedimiento unario P a los enteros desde I1 hasta I2 incrementado en I3.

- Restricción 7:

```
{For 1 NumArticulos 1
       proc {$ I}
         {For 1 NumEvaluadores 1
                proc {$ J}
                  if {Bool.'not' {List.member IdiomaArticulo.I IdiomaEvaluadores.J}}
                  then
                     Dom.I \=: J
                  end
                end
         }
       end
}
```

Por medio de la instrucción **if**, que aparece en el código anterior, se está determinando si el idioma del artículo I no corresponde a los idiomas que maneja el evaluador J; de ser así, se impone que al artículo I no se le puede asignar el evaluador J (esto se indica por medio de Dom.I `\=:` J). El anterior proceso se realiza para cada uno de los evaluadores y para cada uno de los artículos, utilizando dos ciclos anidados **FOR**.

- Restricción 5:

```
{For 1 NumArticulos 1
       proc {$ I}
         {For 1 NumEvaluadores 1
                proc {$ J}
                  if PaisArticulo.I == PaisEvaluador.J then
                     Dom.I \=: J
                  end
                end
         }
       end
}
```

Por medio de la instrucción **if**, que aparece en el código anterior, se está determinando si el país del artículo I es el mismo país que el del evaluador J; de ser así, se impone que al artículo I no se le puede asignar el evaluador J (esto se indica con Dom.I `\=:` J). El anterior proceso se realiza para cada uno de los evaluadores y para cada uno de los artículos, utilizando dos ciclos anidados **FOR**.

6. ANÁLISIS DE LOS DOS MODELOS

En esta sección, se presenta un análisis de los dos modelos mencionados. Se consideraron, además de la definición misma de los modelos, los tiempos de respuesta obtenidos con su implementación. Se debe tener en cuenta que aunque en la implementación de una solución basada en restricciones uno de los factores de eficiencia corresponde a la estrategia de distribución, la estrategia empleada en este caso, para los dos modelos, es genérica y equivalente. Por lo tanto, los resultados de eficiencia dependen, en gran medida, del modelo que se utilice (y no de su implementación) y por ello fueron utilizados para su comparación.

Para las pruebas se han considerado datos reales que corresponden al evento académico **CLEI** 1996, el cual se realizó en Bogotá, Colombia.

6.1. Consideraciones del Modelo en un Lenguaje de Programación por Restricciones

La Programación por Restricciones trata con problemas de optimización[5] usando la idea de verificar la satisfactibilidad de un conjunto de restricciones encontrando un testigo de ello. Este testigo, además de ser una solución al problema, debe satisfacer él mismo una restricción que lo obliga a tener un costo mejor que el que se haya encontrado hasta el momento (inicialmente, cualquier testigo sirve). Cuando no se encuentran nuevos testigos, el último corresponde a una solución óptima. Normalmente este proceso es controlable, y

[5] Uno de ellos es el que se presenta en este artículo

los usuarios pueden detenerlo cuando la solución encontrada es suficientemente buena a su criterio, aunque no se pueda asegurar que sea la óptima.

De acuerdo a lo mencionado en la sección 2, la ejecución de un programa CCP consiste de un conjunto de agentes (encargados de satisfacer una restricción básica específica) podando los posibles valores de las variables (etapa llamada de propagación). Cuando ninguno de los agentes puede podar más, se pasa a la etapa de distribución, la cual a su vez da lugar a más etapas de propagación. El número de variables, el número de restricciones y la capacidad de poda de cada una, son factores fundamentales en la eficiencia total del proceso y están directamete relacionadas con el modelo mismo. También es un factor de eficiencia, la estrategia de distribución.

En el modelo binario, existen $numEvaluadores * numArticulos$ variables 0-1 ($Dom_{i,j}$). Además, se usa otra variable que indica el número de evaluaciones asignadas (**C**); lo ideal es que (**C**) corresponda con el número total de evaluaciones deseadas. Sin embargo ésto podría no ser posible; por ello se busca el mayor valor de (**C**) con el que se puedan satisfacer todas las restricciones.

Para la comparación, la estrategia de distribución utilizada, distribuye las variables en dos momentos. En el primero, se distribuye la variable **C**, partiendo de los valores más altos. En el segundo se distribuyen las variables correspondientes a los artículos, $Dom_{i,j}$, en el órden en que se reciben e iniciando, en cada una de ellas, con el valor más alto del dominio(valor = 1).

Los distribuidores en **MOzArt**, para el modelo binario, se escriben así:

$$\{FD.distribute \;\; generic(value : max) \;\; [C]\}$$

$$\{FD.distribute \;\; generic(order : \;\; naive \;\; value : \;\; max) \;\; Dom\}$$

En el modelo alternativo, existen $numArticulos{*}numEvaArt$ variables ($Dom_{i,k}$), con dominio $[0..numEvaluadores]$ (0 significa que no se encontró evaluador; en otro caso significa el número del evaluador asignado). Además, se usa otra variable (**C**), que juega el mismo rol que en el modelo binario.

Al igual que en el modelo binario, se distribuyen las variables en dos momentos, distribuyendo en el primero la variable **C** de la manera descrita para el modelo binario. En el segundo se distribuyen las variables correspondientes a los artículos, $Dom_{i,k}$; primero se distribuyen aquellas variables con dominio más restringido (primero asignar evaluadores a los artículos con menor número posible de evaluadores); el orden en que se escogen los valores posibles puede ser cualquiera (en este caso se escogió el mayor valor primero).

Los distribuidores en **MOzArt**, para el modelo alternativo, se escriben así:

$$\{FD.distribute \;\; generic(value : max) \;\; [C]\}$$

$$\{FD.distribute \;\; generic(order : \;\; min \;\; value : \;\; max) \;\; Dom\}$$

Como se puede observar, las estrategias de distribución escogidas para cada modelo son genéricas y equivalentes. Por tanto la eficiencia del proceso está directamente ligada a los modelos propuestos.

6.2. Resultados Obtenidos

A continuación se presentan los resultados obtenidos considerando problemas de diferentes tamaños y utilizando ambos modelos, pero utilizando siempre el mismo conjunto de datos (del CLEI 1996).

En los cuadros 3 y 4 se presentan tiempos obtenidos en la búsqueda de soluciones parciales al problema, considerando diferentes valores para **C**. Se considera el tiempo de búsqueda de aquellas soluciones que tengan por lo menos **C** evaluaciones asignadas.

Calidad de la Solución (C)	**T. Mod. Alt. (Seg)**	**T. Mod. Bin. (Seg)**
50	0.210	2.3
100	0.700	10.5
110	1.10	15.4
121 (óptimo)	2.23	42.52

Cuadro 3: Soluciones obtenidas por los dos modelos para el problema con 44 artículos y 12 evaluadores

Como se puede apreciar en los cuadros 3 y 4, el tiempo que demora el programa en encontrar una solución se ve afectado por el número de evaluaciones (**C**) que se esperan. Por ejemplo al, utilizar el modelo binario con el problema de 80 artículos y 20 evaluadores no se encontró solución óptima después de 3 horas. Cuando el número de evaluaciones esperado es pequeño, la respuesta se obtiene rápidamente. A medida que el número de evaluaciones esperado se incrementa, también los tiempos de respuesta aumentan, hasta

Calidad de la Solución (C)	T. Mod. Alt. (Seg)	T. Mod. Bin. (Seg)
100	3.23	20.8
140	4.578	65.1
180	23.4	227.1
221 (óptimo)	40.265	—

Cuadro 4: Soluciones obtenidas por los dos modelos para el problema con 80 artículos y 20 evaluadores

alcanzar el máximo tiempo de respuesta al momento de encontrar la solución óptima. Esto se debe a que a medida que la calidad que se exige aumenta, las soluciones en el espacio de búsqueda que se ajustan se reducen, y por lo tanto son más difíciles de encontrar.

En el cuadro 5 se presentan los resultados obtenidos al buscar la solución óptima utilizando ambos modelos en problemas de diferente tamaño.

Probl.	T. Mod. Alt.	# Nod. Mod. Alt.	T. Mod. Bin.	# Nod. Mod. Bin.
28 art. y 8 eval.	0.020	414	32	369527
80 art. y 20 eval.	40.265	17373	—	Sup. a 2000000
120 art. y 25 eval	110.34	43897	—	Sup. a 2000000
180 art. y 25 eval.	325.45	76561	—	Sup. a 2000000

Cuadro 5: Resultados comparativos considerando diferentes problemas

Observando los tiempos de respuesta obtenidos en las diferentes pruebas realizadas sobre diversos datos, se puede apreciar que el modelo alternativo encuentra una solución óptima en menor tiempo que el modelo binario.

El modelo alternativo supera en tiempo al modelo binario. Primero que todo porque en la representación del problema, las variables de decisión del modelo alternativo corresponden a un espacio de búsqueda de tamaño inferior al tamaño del espacio de búsqueda del modelo binario.

El segundo, sin propagación, es de $2^{numArticulos*numEvaluadores}$. El primero, por su lado, es en el peor caso (todos los evaluadores evalúan de todos los temas) de $numEvaluadores^{(numArticulos*numEvalArt)}$.

Si se tienen 180 artículos, 25 evaluadores y 3 evaluaciones por artículo (como en el caso del clei 1996), el tamaño del espacio de búsqueda del modelo binario es 2^{180*25}, que es aproximadamente 2^{1800} **veces más grande** que el espacio del modelo alternativo, 25^{540}.

7. CONCLUSIONES Y TRABAJOS FUTUROS

En este trabajo se presenta el uso de la programación por restricciones en un problema de optimización combinatoria real, como lo es el de la asignación de evaluadores a artículos.

Se propusieron dos modelos y se estableció que el modelo alternativo fue el más eficiente respecto al tiempo. Con ambos modelos se obtuvieron soluciones para problemas de tamaño pequeño y mediano, dadas las fortalezas que presenta la programación por restricciones para este tipo de problemas.

Con el modelo alternativo se obtuvieron los mejores tiempos, esto se debe en gran parte, a la representación de las variables de decisión. Debido a esta representación el espacio de búsqueda del modelo alternativo es más pequeño al del modelo binario.

Utilizando la implementación del modelo alternativo, se obtuvieron resultados en corto tiempo para problemas de tamaño medio y grande, incluyendo el problema de la distribución de evaluadores para artículos en el **CLEI** 1996.

La eficiencia obtenida utilizando un lenguaje de programación por restricciones depende del modelo utilizado y de la estrategia de distribución. Con base en lo anterior, se han propuesto algunas estrategias de distribución basadas en heurísticas orientadas al problema en particular, por medio de las cuales se pueden obtener soluciones en menor tiempo. Sin embargo, el interés de este artículo ha sido el de mostrar unos modelos y analizarlos, y por ello se han utilizado estrategias de distribución independientes del problema. En otro artículo, se presentará un estudio de las estrategias de distribución basadas en heurísticas.

En este trabajo se pudo observar lo fundamental que resulta el modelaje de un problema de optimización combinatoria en la obtención eficiente de resultados.

Se espera desarrollar una herramienta robusta que sirva a diversos eventos académicos en la toma de decisiones acerca de la distribución de los artículos. Adicionalmente se espera acoplar la herramienta para

que haga parte de sistemas de apoyo existentes como **WIMPE** [6] u **OpenConf** [10].

Referencias

[1] C. Castro and S. Manzano. Variable and value ordering when solving balance academics curriculum problem. **In: Proc. of the ERCIM WG on constraints**. 2001.

[2] Juan F. Díaz and Camilo Rueda. VISIR: Software de soporte para la toma de decisiones de vertimiento de agua en la represa del alto anchicayá usando programación concurrente por restricciones. **Ingeniería y Competitividad**. Vol 3, No. 2, 2001.

[3] M. Henz and M. Muller. Programming in Oz. *In G. Smolka and R. Treinen, Editors, DFKI Oz, Documentation Series*. 1995.

[4] Brahim Hnich, Zeynep Kiziltan and Toby Walsh. Modelling a Balanced Academic Curriculum Problem. **In: Proceedings of CP-AI-OR-2002**. 2002.

[5] K. Marriot and P. J. Stuckey. Programming with Constraints: An Introduction. *MIT Press*, Cambridge, Mass, 1998.

[6] David M. Nicol. WIMPE: Web Interface for Managing Programs Electronically. http://www.crhc.uiuc.edu/ñicol//wimpe/wimpe6.1.html. 2001.

[7] C.Rueda and J.F. Díaz and L.O. Quesada and C. García and S. Cetina. PATHOS: Object Oriented concurrent constraint timetabling for real world cases. In Proceedings **XXVIII Conferencia Latinoamericana de Informática**, Montevideo, Uruguay, Noviembre de 2002.

[8] Christian Schulte and Gert Smolka. Finite Domain Constraint Programming in OZ. A Tutorial. **MOzArt Documentation**, 2004.

[9] P. Van Roy and Seif Hairidi. Concepts, Techniques, and Models of Computer Programming. *MIT Press*, 2004.

[10] Zacon Group. OpenConf-Conference Manual Management System. http://www.OpenConf.org. 2004.

Um modelo para Certificação ISO 9001:2000 em PMEs

Raimundo S.N. Azevedo
Arnaldo D. Belchior
Universidade de Fortaleza, Mestrado em Informática Aplicada,
Fortaleza-Ceará, Brasil, 60811-341
sales.mia@unifor.br, belchior@unifor.br

and

Marum S. Filho
Flávio L. César
Softexport, Diretoria,
Fortaleza-Ceará, Brasil, 60811-341
marum@softexport.com.br, lenz@softexport.com.br

Abstract

This work describes a model for certification ISO 9001:2000 process of a software factorty. The Quality Management System (QMS) implementation in the factory guided the *fundamental processes* of the software development, the *organizational processes* and the *support processes*, structuring all the practices in the company and extending its vision in relation to their own products, processes, collaborators and clients. The certification process was conducted through a set of structured activities based upon ISO/IEC 12207, in contrast to the logical sequence of implementation items in ISO 9001. As a result, this work provided greater agility in the implementation process of the QMS in the software factory.
Keywords**:** *software process, ISO certification, ISO 9001, ISO/IEC 12207, ISO/IEC 15271*

Resumo

Este trabalho descreve um modelo para o processo de certificação ISO 9001:2000 de uma fábrica de software classificada como PME. A implantação do Sistema de Gestão da Qualidade (SGQ) na fábrica passou a orientar os *processos fundamentais* de desenvolvimento de software, os *processos organizacionais* e os *processos de apoio*, estruturando todas as rotinas da empresa e ampliando a visão da mesma em relação a seus próprios produtos, processos, colaboradores e da relação com seus clientes. O processo de certificação foi conduzido através de um conjunto de atividades estruturadas a partir da ISO/IEC 12207, diferenciando-se da seqüência lógica de implantação dos itens da ISO 9001. Isto deu uma maior agilidade no processo de implantação do SGQ.
Palavras-chave: processo de software, ISO 9001, ISO/IEC 12207, ISO/IEC 15271.

1. Introdução

A certificação NBR ISO 9000 [1] é reconhecida em todo o mundo em vários setores produtivos. A conquista da certificação ISO leva a organização a um nível de padrão internacional de qualidade em seus processos. No entanto, no âmbito de um determinado setor, não é possível diferenciar o nível de maturidade de uma empresa em relação a outra, em um conjunto de empresas que tenham recebido a certificação ISO 9000, a não ser pelo escopo de certificação, pela qualidade do certificador e pelo tempo pelo qual a certificação vem sendo mantida [5, 6].

A ISO 9000 é um padrão internacional de qualidade, que aplica o gerenciamento da qualidade do processo para gerar produtos, que atentam às expectativas de seus usuários. Esses padrões foram criados sob a premissa de que, se o desenvolvimento e o gerenciamento do sistema são de boa qualidade, então, o produto ou o serviço resultante também será de boa qualidade. Um sistema de Gestão da qualidade em conformidade com a ISO 9000 assegurará que seu processo de desenvolvimento possui um nível de controle, disciplina e repetibilidade, garantindo a qualidade de seus produtos.

No Brasil, segundo dados retirados de Paduan [3], ao final do ano 2000 existiam 5.400 empresas ligadas à produção e comercialização de Software com um total de 158.000 funcionários. Fazendo-se um cálculo simples, observamos que, em média, temos cerca de 29 funcionários por empresa, Isto constata uma maioria de pequenas e médias empresas nacionais na área de desenvolvimento de software. Para isto, consideram-se pequenas as empresas com até 49 funcionários, e médias as com até 99 funcionários.

Porte da empresa	Faixa
Micro	De 1 a 9 pessoas
Pequena	De 10 a 49 pessoas
Média	De 50 a 99 pessoas
Grande	A partir de 100 pessoas

Tabela 1. - MCT/Qualidade e Produtividade no Setor de Software Brasileiro - 2001

Aliado a este número, temos as empresas cujo desenvolvimento de software não é uma atividade fim. Portanto, o software gerado não tem característica de comercialização e sim de uso interno.

Em ambos os setores mencionados, pequenas e médias empresas (PME) e empresas de desenvolvimento interno (EDI), não há justificativas para não se perseguir o desenvolvimento de produtos de software de qualidade, com procedimentos detalhados e documentados, registros, acompanhamentos, dados históricos gerados e regidos por uma política institucional, que seja seguida por todos, e que evolua com a maturidade dos próprios processos e portanto implantar um SGQ que ligue os aspectos operacionais da garantia da qualidade com seus aspectos gerenciais e estratégicos.

Na PME, temos algumas características:

- A diretoria acumula papéis nos processos de desenvolvimento e tem de assumir os papeis institucionais administrativos.
- Pouca disponibilidade de recursos financeiros;
- Alta rotatividade dos colaboradores;
- Acúmulo de papéis e responsabilidades;
- Papel formador de mão-de-obra especializada;
- Clientes cada vez mais exigentes quanto a prazos, custos e qualidade do produto;
- A existência de produtos de software desenvolvido internamente com potencial comercial;

Muitas vezes, esta reduzida estrutura organizacional se choca com os modelos de maturidade, os quais apontam para uma estrutura pesada e densamente povoada por inúmeras funções e responsabilidades [8]. Na EDI, em geral, não existe o interesse direto por uma certificação, e sim pelos efeitos da implantação de um Sistema de Gestão da Qualidade (SGQ) na redução de seus custos e na qualidade de seus produtos de software.

A abordagem genérica das normas ISO 9000 e ISO 9001[2] faz com que o sistema de gestão da qualidade proposto, seja aplicado em qualquer organização possuidora de processos que recebam insumos como entrada tendo como resultado produtos intermediários ou finais. Esta abordagem, no entanto, as tornam genéricas demais para a aplicação em pequenas e médias empresas de desenvolvimento de software, onde grande parte dos componentes da organização, pessoas, processos e atividades, estão diretamente ligados à rotina de produção de software. Quando a organização tem um histórico de produtos e clientes, os processos e metodologias de desenvolvimento de alguma forma já existem e são aplicadas. As pessoas que participam da empresa, a diretoria, os desenvolvedores, os

analistas de sistemas, analistas de teste, os gerentes de projeto, os designers e o suporte convivem diariamente com as rotinas do desenvolvimento. A grande dificuldade está no entendimento da visão organizacional por esse grupo. As rotinas administrativas e financeiras devem ser atendidas e não são rotinas simples.

A abordagem da Norma NBR ISO/IEC 12207[4] e sua descrição do ciclo de vida do software, aproxima o ambiente da organização e suas atividades diárias de desenvolvimento de software com os conceitos de sistemas, processos, atividades e tarefas de uma maneira clara para todos na organização.

A proposta deste trabalho é a adoção da estrutura da norma ISO 9001:2000 (ISO 9001), para a estruturação do Sistema de Gerenciamento da Qualidade dos processos, para gerar produtos, e para a criação, ou organização, da cultura da qualidade em pequenas e médias empresas utilizando a estrutura descrita na Norma ISO 12207.

A percepção de um padrão é um indicador que houve um entendimento. O elemento analisado de alguma forma se encaixou em um comportamento já conhecido e de domínio do agente analisador. O padrão da ISO 12207, conhecido e de vocabulário aplicado nas rotinas de desenvolvimento de software, se integra com o padrão da ISO 9001, genérico, abrangente e de certa forma distante do dia-a-dia da organização em sua aplicação. Segundo Campos[9], o termo padrão refere-se a tudo que se unifica e simplifica para o benefício das pessoas, adotado por consenso e podendo ser alterado, incluindo-se nesse caso, procedimentos, conceitos e também métodos de medida.

2. As Normas NBR ISO 9000:2000 e NBR ISO 9001:2000

A ISO 9000 [1] e a ISO 9001 [2] foram elaboradas para orientar as organizações na implementação eficaz de seu *sistema de gestão da qualidade* (SGQ). A partir da implementação desse sistema, cujo foco é a garantia da qualidade, a empresa estará preparada para tratar com maior agilidade e eficácia a ocorrência de problemas. A qualidade vai sendo complementada em seus processos e produtos, em uma ação de melhoria contínua.

A ISO 9000 define os fundamentos e o vocabulário que envolvem e definem o SGQ. Para a gerência e o controle de uma organização, a alta direção tem de ter uma visão sistêmica de todos os processos e atividades, identificando os pontos de influência e de alcance de toda a estrutura organizacional, envolvendo de forma integral clientes e fornecedores nos processos institucionais. Em apoio a essas necessidades a ISO 9000 identifica oito princípios para a gestão da qualidade:

- *Foco no cliente;*
- *Liderança;*
- *Envolvimento de pessoas;*
- *Abordagem de processo;*
- *Abordagem sistêmica para a gestão;*
- *Melhoria contínua;*
- *Tomada de decisão baseada em dados e informações;*
- *Benefícios mútuos nas relações com os fornecedores;*

A aplicação integrada desses oito princípios necessita de várias etapas para o desenvolvimento e a implementação do SGQ, são estas:

- *Determinação das necessidades e expectativas dos clientes e outras partes interessadas;*
- *Estabelecimento da política da qualidade e dos objetivos da qualidade da organização;*
- *Determinação dos processos e responsabilidades necessários para atingir os objetivos da qualidade;*
- *Determinação e fornecimento dos recursos para atingir os objetivos da qualidade;*
- *Estabelecimento de métodos para determinar a eficácia e a eficiência de cada processo;*
- *Aplicação dessas medidas para determinar a eficácia e a eficiência de cada processo;*
- *Determinação de meios para prevenir não-conformidades e eliminar suas causas;*
- *Estabelecimento e aplicação de um processo para melhoria de um sistema de gestão da qualidade.*

No seguinte trecho transcrito da Norma temos outro enfoque nos processos:

"Para que as organizações funcionem de forma eficaz, elas têm que identificar e gerenciar processos inter-relacionados e interativos. Freqüentemente, a saída de um processo resultará diretamente na entrada do processo seguinte. A identificação sistemática e a gestão dos processos empregados na organização e, particularmente, as interações entre tais processos são conhecidos como 'abordagem de processos'". Essa abordagem está representada na figura 1.

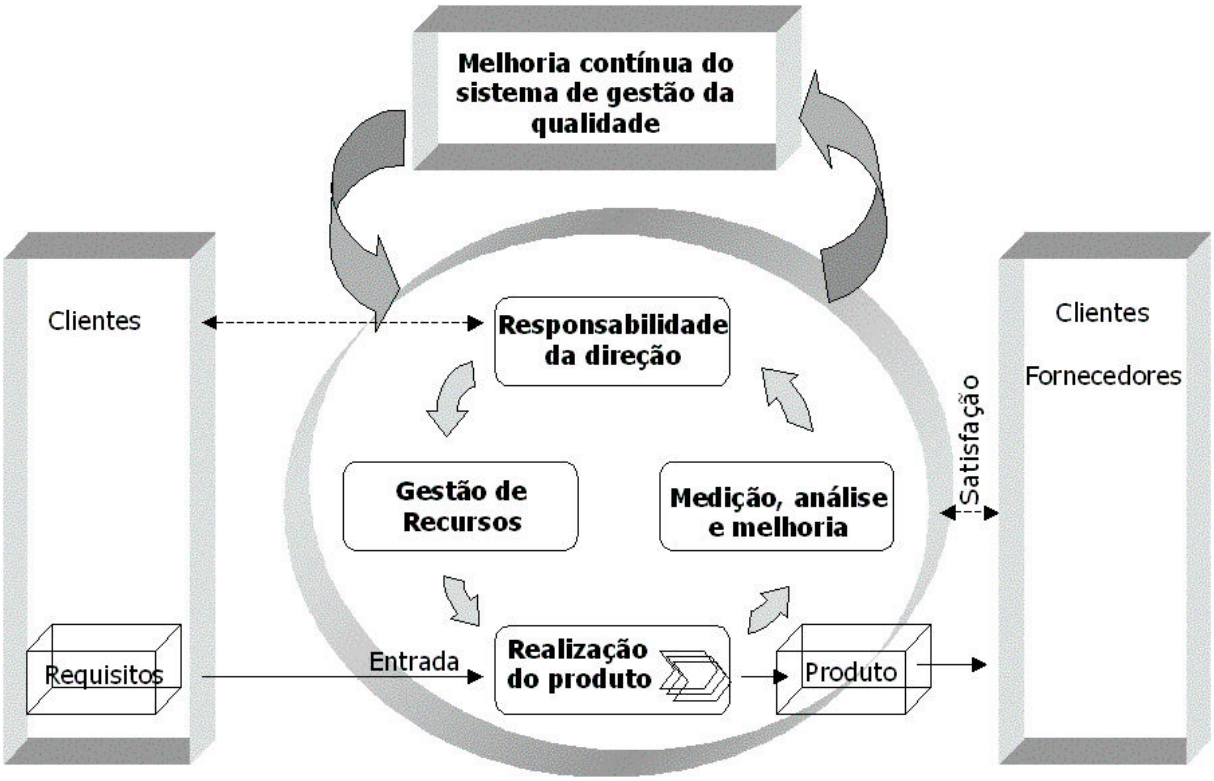


Figura 1. Modelo de um processo baseado no sistema de gestão da qualidade

O sistema de gestão da qualidade de uma empresa possui um conjunto de diretrizes, que permite a seus clientes avaliar a capacidade dessa organização em fornecer produtos e serviços, que atendam os requisitos especificados de forma consistente, fornecendo ainda uma estrutura para melhoria contínua do desempenho da organização. Uma empresa, que deseja certificar seu sistema de gestão da qualidade segundo a ISO 9001, deverá considerar as seguintes questões:

- *conhecer e demonstrar sua capacidade em atender os requisitos dos clientes*;
- *planejar e documentar todas as atividade que afetam a qualidade*;
- *qualificar pessoas nas competências necessárias à realização de tarefas*;
- *identificar e disponibilizar recursos materiais e humanos necessários para manter o sistema da qualidade*;
- *registrar a execução das atividades*;
- *prevenir as não-conformidades e, se ocorrerem, devem ser registradas e tratadas*;
- *identificar os processos críticos para a satisfação dos clientes*;
- *manter um programa contínuo de avaliação do desempenho do sistema*.

A organização deve aplicar sua política da qualidade para evidenciar o comprometimento da alta direção para com a qualidade. Deve estar adequada aos propósitos da organização, ser verdadeira e refletir os valores da empresa para todos os clientes, funcionários e demais interessados.

A norma ISO 9001 recomenda que, para um gerenciamento eficaz, a melhor forma de atender aos requisitos é a criação de procedimentos documentados. Os procedimentos são uma forma especificada para desenvolver uma atividade. Ela exige apenas a elaboração de seis procedimentos:

- *Controle de documentos*
- *Controle de registros*
- *Auditoria interna*
- *Controle da não conformidade de produtos*
- *Ação corretiva*
- *Ação preventiva.*

A existência de procedimentos, instruções e registros de trabalho formalizam todas as atividades que afetam a qualidade. Isto exige a participação de todos os indivíduos da organização. Portanto, a conscientização para com a qualidade aumenta, uma vez que todos participam diretamente da implementação do sistema da qualidade, pois são os principais responsáveis pelas atividades da empresa.

O cliente influencia o início, o próprio processo e o produto final. O domínio de um procedimento de desenvolvimento de software e do gerenciamento do SGQ leva ao domínio do processo de produção, o que pode garantir uma qualidade do produto final. O SGQ é o apoio para a conformidade dessa norma, e deve estar estruturado para controlar e divulgar o conjunto dos procedimentos usados pela empresa, facilitar e promover o gerenciamento de mudanças e facilitar as atividades de monitoração e de auditorias do sistema da qualidade.

A empresa para se certificar segundo a norma ISO 9001 deve conhecer e mostrar que entende os requisitos do cliente, planejar e documentar as atividades que afetam a qualidade, qualificar pessoas, identificar e disponibilizar recursos para manter o sistema da qualidade, registrar as atividades, prevenir não-conformidades, registrar a ocorrência delas e tratar as causa dessas não-conformidades, além de manter um programa de avaliação contínuo do SGQ, para a melhoria contínua de todos os processos.

3. As Norma NBR ISO/IEC 12207:1998 e NBR ISO/IEC 15271:2000

A ISO/IEC 12207 [4], cuja arquitetura utiliza uma abordagem sistêmica e usa os conceitos de processos (conjunto de inter-relacionadas, que transforma entradas em saídas), atividades (engloba a utilização de recursos) e tarefas (expressa na forma de um requisito, auto-declaração, recomendação ou ação permitida), descreve o ciclo de vida de software em três macro-processos:

- Processos fundamentais: Agrupam as partes que integram diretamente a produção do software, sendo eles: Aquisição, Fornecimento, Desenvolvimento, Operação e Manutenção.
- Processos de apoio: Os processos de apoio auxiliam outros processos na busca do sucesso e da qualidade do projeto e são formados por: Documentação, Gerência de configuração, Garantia da qualidade, Verificação, Validação, Revisão conjunta, Auditoria e Resolução de problema.
- Processos organizacionais: Os processos organizacionais envolvem tipicamente políticas e práticas institucionais, sendo compostos por: Gerência, Infra-estrutura, Treinamento e Melhoria.

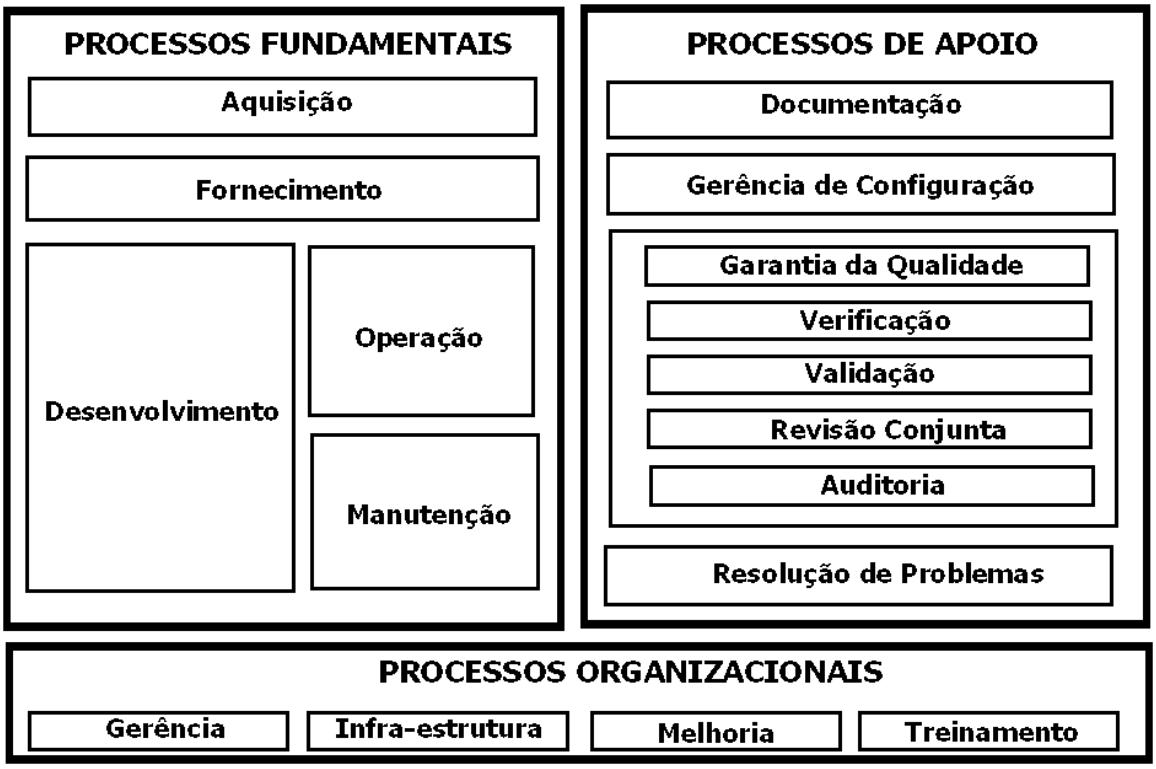


Figura 2. Processos de ciclo de vida se software

Cada macro-processo é composto de processos que são compostos de um conjunto de atividades, que por sua vez são compostas de um conjunto de tarefas. Os processos dessa Norma formam um conjunto abrangente e genérico na sua aplicação. Cada organização pode selecionar o subconjunto apropriado para suas atividades, podendo adaptar inclusive para projetos ou aplicações específicos. O alcance da estrutura da Norma também é apropriado para a utilização quando o software é uma entidade independente, embutida ou integrada a um sistema(Conjunto integrado que consiste em um ou mais processos, hardware, software, recursos e pessoas, capaz de satisfazer uma necessidade ou objetivo definido).

Existem na ISO 12207 os anexos A – (normativo) Processo de adaptação e B - (informativo)Orientação para adaptação, que prevendo as variações nas políticas e procedimentos organizacionais, métodos e estratégias de aquisição, tamanho e complexidade dos projetos, requisitos e métodos de desenvolvimento do sistema sugere que todas as partes envolvidas nos projeto deveriam ser envolvidas nessa adaptação à realidade da organização seguindo as regras e relacionamentos definidos.

Segundo a NBR ISO/IEC 15271 [7], " A NBR ISO/IEC 12207 estabelece uma forte ligação entre o sistema como um todo e o software", "Até certo ponto a NBR ISO/IEC 12207 é projetada para atuar dentro de um processo de engenharia de sistemas. Quando o software é parte de um sistema total, o software é isolado do sistema, produzido, e reintegrado ao sistema. Quando o software constitui todo o escopo de interesse, as tarefas em nível de sistema podem ser tratadas como uma orientação útil."

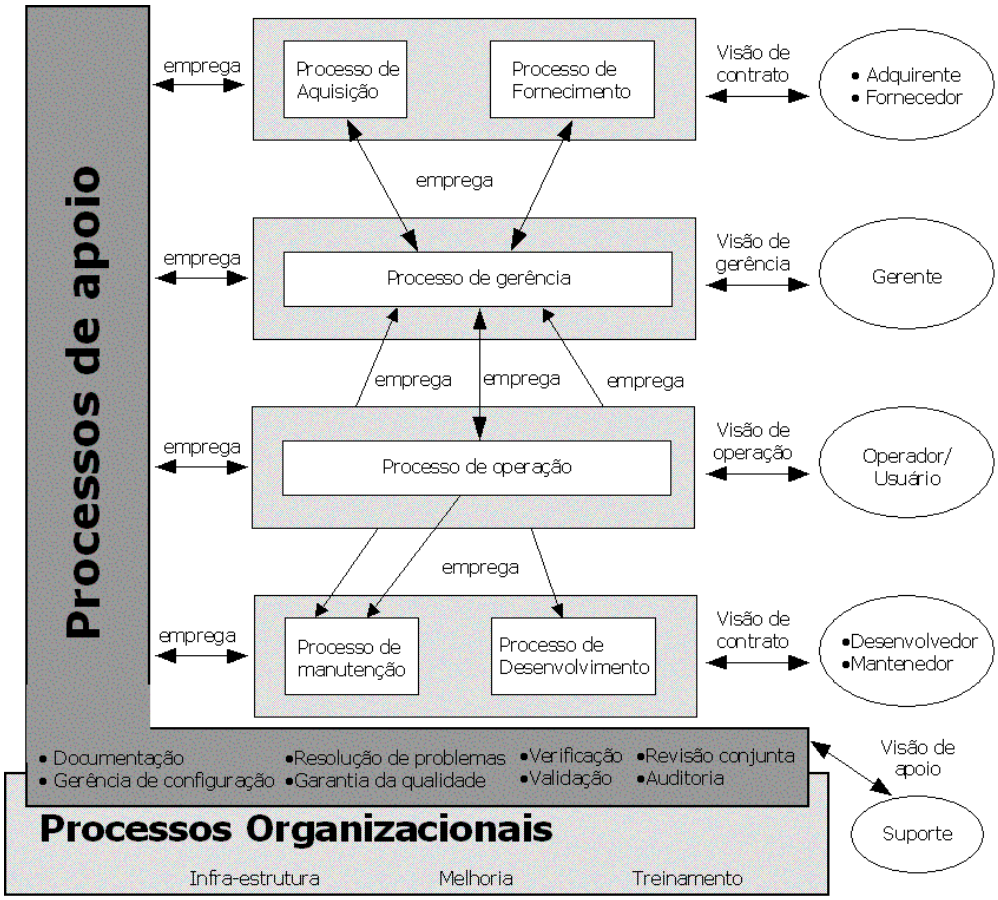


Figura 3. Processos de ciclo de vida de SW - Regras e relacionamentos

Figura 4. Visão do Software no sistema

4. Um modelo para Certificação ISO 9001:2000

O Sistema de Gestão da Qualidade (SGQ), visando centralizar as ações de desenvolvimento de software referentes à qualidade de processos e de produtos, como também ao atendimento das necessidades de seus clientes, deve ser discutido e adotado pela alta administração da empresa. A seguir descrevemos os passos para uma certificação ISO 9001:2000:

- A implantação do SGQ deve ser realizada a partir dos 3 macro-processos da ISO/IEC 12207 [4], que servem como tripé para as atividades de desenvolvimento de software. Esta estrutura atende aos requisitos da ISO 9001 e reflete uma visão sistêmica da empresa e de suas rotinas, tanto as rotinas de produção, quanto as de administração. Com isto, as atividades e tarefas descritas no dia-a-dia da empresa de software, levarão seus colaboradores a identificarem e situarem suas responsabilidades nos seus vários processos formalizados. O mapeamento da ISO 9001 com a ISO/IEC 12207 apoia-se principalmente nesta identificação, possibilitando também à ISO 9001 uma adequação ao perfil técnico de uma fábrica de software. Desta forma, todos os requisitos para a certificação ISO 9001 foram enquadrados nos processos de ciclo de vida do software, além de outros procedimentos importantes e necessários que foram gerados.

- Planejamento da Qualidade deve ser realizado a partir das diretrizes determinadas pela direção da empresa na Política da Qualidade e deve ser coerente com os Objetivos para Qualidade. Nesse planejamento, devem ser levados em consideração os requisitos de gestão da qualidade e a implementação de melhorias contínuas necessárias para atender à Política da Qualidade e manter a integridade do SGQ. Essas melhorias serão operacionalizadas mediante a implementação de mudanças no SGQ, de forma a mantê-lo permanentemente atualizado e em consonância com os requisitos especificados pelos clientes.

- As atividades que influenciam direta ou indiretamente a qualidade, serão realizadas de forma estruturada. Essa estrutura inclui atividades de verificação e de controle da qualidade nas diversas etapas dos processos, as quais serão realizadas de forma planejada e obedecendo a requisitos e instruções definidas nos procedimentos documentados do SGQ.

- SGQ deve ser mantido atualizado e sua eficácia será melhorada de forma contínua, de acordo com o estabelecido na Política da Qualidade e pelas diretrizes básicas da superior administração. Suas características devem estar descritas em procedimentos documentados, os quais serão elaborados de forma a descrever a metodologia de realização das atividades relacionadas com a qualidade de processos e produtos. A estrutura documental do SGQ é composta por:
  - Leis, decretos, portarias (Documentos legais)
  - Manual da Qualidade (Documento Principal do SGQ)
  - Procedimentos (Documentos Normalizadores)
  - Tutorais (Documentos Operacionais)
  - Formulários/Modelos (Suporte para registros de dados)
  - Registros (Resultados / evidências)

- Manual da Qualidade deve incluir: a descrição do escopo de certificação, as políticas da qualidade, referência aos procedimentos documentados usados nas diferentes atividades relacionadas com os processos administrativos e produtivos, e descrição da interação entre os processos do SGQ.

- As atividades que influenciam direta ou indiretamente a qualidade dos processos ou produtos, devem ser descritas nos Procedimentos e Tutoriais, que serão utilizados para assegurar a correta realização das mesmas e o atendimento aos requisitos especificados. No SGQ da fábrica de software, devem existir pelo menos 13 procedimentos documentados:
  - *Procedimentos organizacionais*: contribuem para a melhoria da fábrica e estão fora do domínio de um único projeto. São eles:
    - Procedimento para segurança;
    - Procedimento de responsabilidades da direção;
    - Procedimento para qualificação de fornecedores;
    - Procedimento de qualificação profissional;
    - Programa Estratégico de Qualificação.
  - *Procedimentos de apoio*: auxiliam e contribuem para o sucesso e a qualidade do projeto/produto.
    - Procedimento de controle de documentos;
    - Procedimento de controle de registros do sistema da qualidade;

- Procedimentos para auditorias internas da qualidade;
- Procedimento de controle de produto não conforme;
- Procedimento para ação corretiva, ação preventiva e de melhoria;
- Procedimento para descrever o SGQ.

- *Procedimentos Fundamentais*: usados diretamente no ciclo de vida do software.
  - Procedimento de projeto e desenvolvimento de sistemas;
  - Metodologia de Desenvolvimento de Software adotado pela empresa.

Os procedimentos determinam (ou fazem referência), quando necessário, à utilização de documentos operacionais, técnicos e/ou para suporte de registros e dados, tais como tutorais e formulários. Os Registros da qualidade necessários devem ser definidos nas diferentes fases operacionais, assim como os formulários aplicáveis como suporte desta informação e dos dados. A responsabilidade pela gestão dos processos descritos nos procedimentos documentados está definida no próprio corpo de cada documento, exceto no caso de formulários, onde não é requerido. Quando necessária, a implementação dos documentos do SGQ é precedida de treinamento das funções envolvidas, de forma a garantir a correta interpretação e aplicação dos requisitos especificados.

4.1 Roteiro das Atividades

Os trabalhos para implantação do SGQ e a posterior certificação da empresa são divididos em atividades desenvolvidas pelos participantes, e estão relacionadas na *Tabela 2*. As atividades dessa tabela estão ordenadas de forma a representar a seqüência no processo de certificação ISO 9001:2000 de uma PME. Essas atividades, para uma melhor compreensão, ainda são caracterizadas no seguinte modo:

- *Local*: corresponde à seção da ISO 9001 onde a atividade está localizada. A atividade não prevista na ISO, mas considerada necessária, é marcada com "*N*".
- *Modo*: atividades seqüenciais (*S*) ou desenvolvidas de forma paralela (*P*).
- *Tipo*: atividade executada uma única vez (*U*), ao longo do processo de certificação de forma contínua (*C*), executada de forma secionada (*D*), e repetidas vezes (*R*).

Tabela 2: Atividades do projeto SGQ

ATIVIDADES	**Local**	**1. Modo**	**Tipo**
Abertura dos trabalhos, apresentação do projeto SGQ.	*N*	*S*	*U*
Levantamento dos processos e produtos da empresa.	4	*P*	*D*
Estudo dos processos de desenvolvimento da empresa.	4	*P*	*D*
Estudo dos produtos de software oferecidos pela empresa.	4	*P*	*D*
Estudo da norma ISO 9000:2000.	*N*	*P*	*D*
Definição da política de qualidade da empresa: Missão, Visão do Futuro, Valores e Princípios.	4	*S*	*U*
Objetivos para a qualidade: Definir objetivos, Estabelecer metas, Estabelecer prazos, Estabelecer responsáveis, Estabelecer padrões e indicadores de desempenho.	4	*S*	*U*
Elaborar procedimentos documentados obrigatórios requeridos pela norma: Controle de documentos, Controle de registros, Controle de produtos em não-conformidade, Controle de ações corretivas, Controle de ações preventivas, Controle de auditorias internas.	4	*S*	*U*
Elaborar procedimentos documentados não obrigatórios responsáveis pelo processo de desenvolvimento de software.	*N*	*S*	*U*
Identificar processos da organização na ótica da norma.	*N*	*P*	*C*
Identificar os registros necessários e implementar os controles requeridos.	4	*S*	*D*
Definir e alocar recursos necessários.	6	*S*	*U*
Identificar os requisitos de competências necessários para as funções que afetam a qualidade dos processos.	6	*S*	*U*
Verificar necessidades de competências nos recursos humanos.	6	*S*	*U*
Identificar treinamento necessário.	6	*S*	*U*
Verificar eficácia do treinamento.	6	*S*	*U*
Verificar necessidades de recursos financeiros, infra-estrutura e meio ambiente.	6	*S*	*U*
Implementação dos processos relacionados com o produto e com o desenvolvimento de software.	7	*P*	*C*
Análise crítica pela alta administração.	7 e 5	*S*	*D*
Validação e comprometimento da alta administração.	5	*S*	*D*
Auditorias internas da qualidade.	8	*S*	*R*
Processos de melhoria.	8	*P*	*C*
Ações corretivas / preventivas.	8	*P*	*R*
Ajustes finais.	8	*P*	*U*

As seguintes atividades também devem ser desenvolvidas na organização para a conclusão de seu processo de certificação ISO 9001:

- Seleção do órgão certificador.
- Pré-auditoria (presença do certificador e empresa).
- Auditoria de certificação (presença do certificador e empresa).

Além dos procedimentos documentados citados anteriormente, um dos produtos gerados e validados frutos do processo de certificação foi a elaboração de uma metodologia de desenvolvimento de software para a fábrica de software.

5. Conclusão

Este trabalho apresenta o modelo usado no processo de certificação da norma de qualidade ISO 9001:2000 [2] de uma fábrica de software brasileira, a SoftExport, uma PME com aproximadamente 21 colaboradores incluindo os 3 diretores que assumem papeis de desenvolvimento. O SGQ implantado a partir da ISO/IEC 12207 [4] tem assegurado que seu processo de desenvolvimento de produtos tenha um nível de controle, disciplina e repetibilidade, garantindo a qualidade de seus produtos de software gerados. O processo de certificação foi conduzido através de um conjunto de atividades estruturadas a partir da ISO/IEC 12207, diferenciando-se da seqüência lógica de implantação dos itens da ISO 9001, levando a uma maior agilidade no processo de implantação do SGQ na fábrica de software.

Podemos citar como principais lições aprendidas ao longo do processo de certificação ISO 9001:2000 [2]:

- O apoio explícito da alta administração da empresa foi de fundamental relevância, uma vez que abriu aos participantes do processo de certificação todas as estruturas da empresa e de seus colaboradores, de forma prioritária;
- As definições de *Visão, Política e objetivos da Qualidade* motivaram a diretoria a definir e acompanhar indicadores, que realmente interessam e motivam o controle da estrutura organizacional;
- O levantamento e o estudo dos processos de desenvolvimento de software adotados pela empresa proporcionaram aos participantes do processo, uma rápida integração com as rotinas e com os desenvolvedores;
- O levantamento e o estudo de produtos de software da empresa identificou que vários desses produtos eram utilizados internamente nos processos de gerência e desenvolvimento. Alguns desses produtos já se adequavam a itens de controle obrigatório da norma;
- A descrição de todos os papeis da estrutura organizacional da empresa, identificando suas responsabilidades, atividades e qualificação necessárias. Lembrando que um colaborador pode assumir vários destes papéis.
- O estudo da norma ISO 9000:2000 [1] situou o grupo no ambiente de Sistema da Qualidade. O enfoque desenvolvido foi a identificação e a adequação de rotinas, processos , metodologias e produtos da empresa à norma. (Figura 5)
- O mapeamento entre a ISO 9001:2000 [2] e a ISO/IEC 12207 [4] integrou os requisitos de certificação com os processos de ciclo de vida, possibilitando a divisão em três frentes de ação (processos organizacionais, fundamentais e de apoio) e facilitando o diálogo com todos os colaboradores da empresa;
- O seqüenciamento das atividades utilizado no processo de certificação ISO 9001:2000 [2] para a fábrica de software, que foi considerado satisfatório, é apresentado a seguir:
  - *Sistema de Gestão da Qualidade* (*seção 4 da norma*): o estabelecimento, a documentação e a implementação de um sistema de qualidade, descrevendo a necessidade dos processos serem conhecidos e documentados.
  - *Gestão de Recursos* (*seção 6*): apresenta os recursos disponíveis e necessários à prática da qualidade.
  - *Realização do Produto (seção 7)*: mostra a implementação do produto à luz dos procedimentos e requisitos do cliente.
  - *Responsabilidade da Direção (seção 5)*: integra a Alta Administração no próprio processo de implantação do SGQ, validando e adotando, perante todos os colaboradores, o processo da Qualidade.
  - *Medição, Análise e Melhoria (seção 8)*: descreve o processo de medição, análise e melhoria, podendo retornar a qualquer das etapas de implantação.

Desta maneira, o projeto de implantação do SGQ baseado na ISO/IEC 12207 [4] pode ser visto como um processo que segue a ISO 9001 [2] e usa suas atividades de *medição, análise* e *melhoria* (*seção 8*), para ajustar as outras atividades. A atividade de Responsabilidade da Direção assegura a validade e o alcance das propostas e reafirma o apoio necessário a todo o processo.

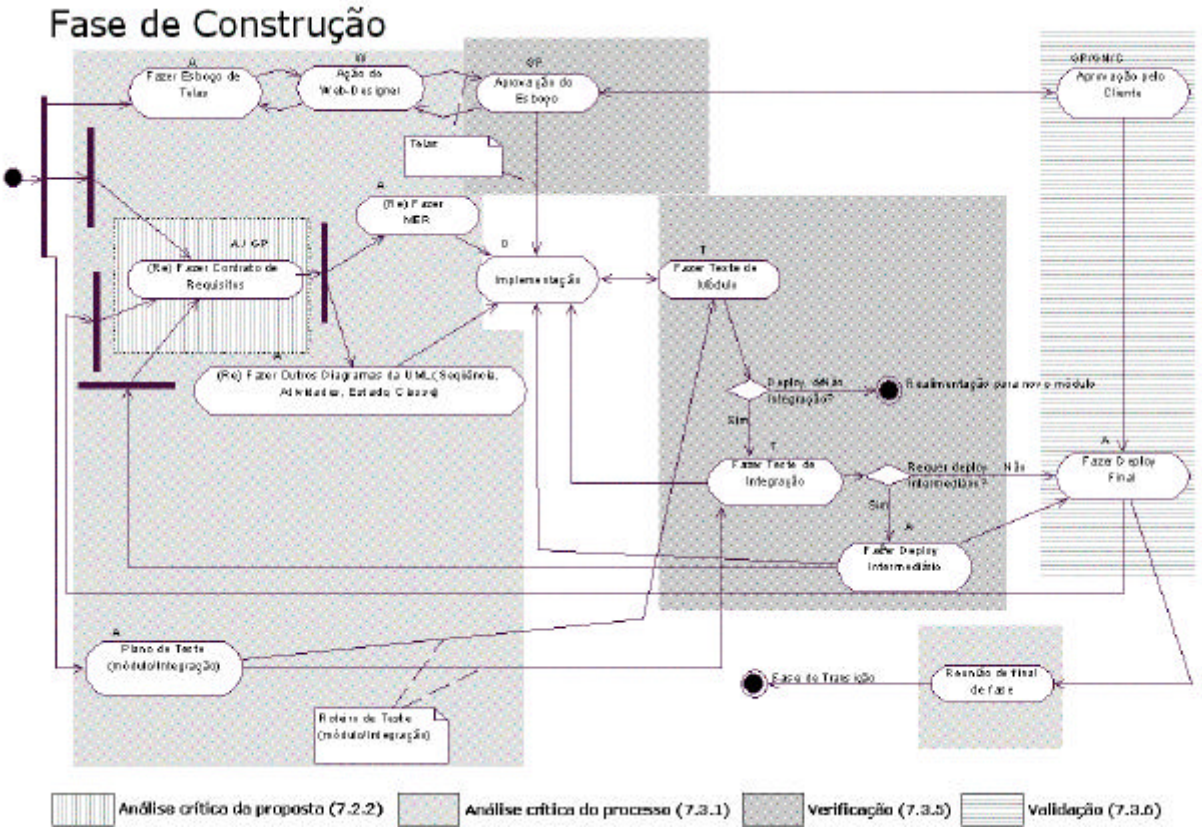


Figura 5. Fase de Construção da metodologia interna indicando requisitos da ISO 9001

Referências

[1] NBR ISO 9000, 2000a, Sistemas de gestão da qualidade – Fundamentos e vocabulário.

[2] NBR ISO 9001, 2000b, Sistemas de gestão da qualidade – Requisitos.

[3] Paduan, Roberta et al. *Tesouro Escondido*. EXAME. São Paulo. v.37, n.13. 2003

[4] NBR ISO/IEC 12207, 1998, *Tecnologia da Informação: processos de ciclo de vida de software*, ABNT, Rio de Janeiro.

[5] Rocha, A. R. C., Maldonado, J. C. e Weber, K. C., 2001, *Qualidade de Software: Teoria e Prática*, São Paulo, Prentice Hall.

[6] Weber. K. C., Rocha,A. R. C. e De Luca, J. C. M, 1999, *Qualidade e Produtividade em Software*, 2ª. Edição, São Paulo, Makron Books.

[7] NBR ISO/IEC 15271, 2000, *Tecnologia da Informação: Guia para ISO/IEC NBR 12207 - processos de ciclo de vida de software*, ABNT, Rio de Janeiro.

[8] Li, Chao et al., *A Software Factory Model on ISO 9000 and CMM for Chinese Small Organizations*, IEEE, 2001.

[9] Campos, Vicente Falconi., 1992, *Qualidade de total: Padronização de empresas*, Belo Horizonte, QFCO.

Yet Another Optimization of the Combinatorial Neural Model

Rafael M. Noivo
Accenture Inc.
Brasília, Brazil, 70.741-640
rafael.moraes.noivo@accenture.com

Hércules A. do Prado
Brazilian Enterprise for Agricultural Research, Embrapa Cerrados,
Brasília, Brazil, 73.301-970
Catholic University of Brasília, Graduate Program in Knowledge and TI Management
Brasília, Brazil, 70.790-160
hercules@{cpac.embrapa.br, ucb.br}

and

Marcelo Ladeira
University of Brasilia, Computer Science Department,
Brasília, Brazil, 70.910-900
mladeira@cic.unb.br

Abstract

Combinatorial Neural Model (CNM) is a classification model that combines both symbolic and connectionist learning approaches. This model is able to recognize regularities from high-dimensional symbolic data, performing mappings from this input space to a set of classes. Due to its hybrid nature, it is possible to extract symbolic relations directly from CNM structure, making it a model of choice for applications that require the rule elicitation. However, this model presents an important drawback: the combinatorial explosion that occurs in its intermediate layer when building the network. To mitigate this problem, CNM has received many modifications that include parallel implementation and relaxation in the building algorithm. In this paper, we describe a new improvement over its architecture that leads to an expressive reduction in the intermediate layer. The model was implemented in the UnBMiner, a framework that provides a large amount of classes for model and data manipulation. An application was developed in the dactyloscopy recognition domain in order to evidence the advantages of our proposal. The space requirement used by our proposal was compared with the usual implementation, and the practical results make clear the gain achieved.
Keywords: Data Mining, Neural Networks, Combinatorial Neural Model, Dactyloscopy Recognition.

Resumo

O Modelo Neural Combinatório (CNM) é um modelo de classificação que combina as abordagens de aprendizado simbólica e conexionista. O modelo é capaz de reconhecer regularidades em dados multidimensionais, realizando mapeamentos entre o seu espaço de entrada e um conjunto de classes. Devido à sua natureza híbrida, é possível se extrair relações simbólicas da estrutura do CNM, o que o torna um modelo interessante para aplicações que requeiram a explicitação de regras. Entretanto, o modelo apresenta uma deficiência importante: a explosão combinatória que ocorre em sua camada intermediária na fase de construção da rede. Para reduzir este problema, o CNM tem recebido diversas modificações que incluem implementações paralelas e relaxamentos no algoritmo de construção. Descrevemos neste artigo uma modificação em sua arquitetura que leva a uma expressiva redução no tamanho da camada intermediária. O modelo foi implementado na plataforma UnBMiner que provê uma grande quantidade de classes para manipulação de dados e modelagem. Uma aplicação no domínio da identificação datiloscópica foi desenvolvida de modo a evidenciar as vantagens da nossa proposta. O requisito de espaço utilizado pela presente proposta foi comparado com a implementação usual e os mesmos resultados práticos obtidos tornaram claros os ganhos alcançados.
Palavras chave: Mineração de Dados, Redes Neurais, Modelo Neural Combinatório, Identificação Datiloscópica.

1. INTRODUCTION

To scale up machine learning algorithms to run over very large databases is an important concern for researchers in the Knowledge Discovery from Databases (KDD) field. In this sense, Combinatorial Neural Model (CNM) [1] has received many improvements ([2], [3], [4] and [5]), that have turned it suitable for KDD. This paper presents an alternative architecture to this model and reports the results achieved when applying it to the dactiloscopy recognition task. The results show that the proposed architecture can lead to a reduced growth rate in the model in comparison to the previous alternative. The implementation was developed in the UnBMiner open source platform. Next section presents a description of CNM. Section 3 explains the proposed architecture. Section 4 describes the dactiloscopy domain in which the application was developed. The last section presents the application and the results obtained.

2. THE COMBINATORIAL NEURAL MODEL

CNM is a hybrid-architecture for intelligent systems that integrates symbolic and connectionist learning paradigms. It has some significant issues, such as the ability to build a neural network from background knowledge; incremental learning by examples, solving the plasticity-stability dilemma [6]; a way to cope with the diversity of knowledge; knowledge extraction of an Artificial Neural Network; and the ability to deal with uncertainty. CNM is able to recognize regularities from high-dimensional symbolic data, performing mappings from this input space to a lower dimensional output space. CNM uses supervised learning and a feedforward topology with three layers: the input layer, the intermediate layer - also called combinatorial - and the output layer (see Figure 1). Each neuron of the input layer corresponds to a concept, which is a complete idea about an object of the domain, expressed by an object-attribute-value form. Their values represent the evidences of the domain application. On the combinatorial layer there are aggregator type neurons, each one connected to one or more neurons of the input layer by fuzzy AND arcs that represent logical concepts. The output layer contains one neuron for each possible class (also called hypothesis), linked to one or more neurons on the combinatorial layer by fuzzy OR arcs that also represent concepts. The synapses may be excitatory or inhibitory and they are characterized by a strength value (*weight*) from zero (not connected) to one (fully connected synapses) that can express the logical relations. For the sake of clarity, we consider only the learning of crisp relations, thus with strength value of synapses equal to one, when the concept is present, and zero, when the concept is not present. The proposed modification does not affect the functions of CNM, since it is applied only in the physical level of the model, having no consequences in the logical level.

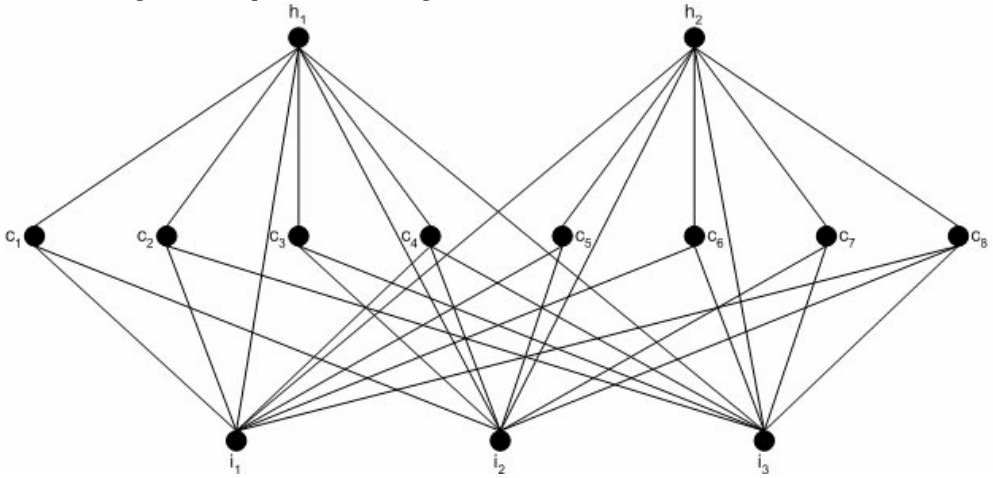


Figure 1 - The complete version of the combinatorial network for 3 input evidences and 2 hypotheses

The network is created completely uncommitted, according to the following steps: a) one neuron in the input layer for each evidence in the training set, b) a neuron in the output layer for each class in the training set, and c) for each neuron in the output layer, there is a complete set of neurons in the combinatorial layer which correspond to all possible combinations of connections with the input layer. This way, each combinatorial neuron represents a combination of connections among two to nine input neurons. This limit in the number of input neuron is a way to avoid combinatorial explosion. This limit is not a heavy restriction; it is only used because of the limitations for processing much information simultaneously [9]. There is no neuron in the combinatorial layer to represent combinations of single input neuron. In this case, each input neuron is connected directly to its hypothesis. The learning mechanism works in a single iteration, and it is described bellow:

PUNISHMENT_AND _REWARD_LEARNING_RULE
• *Set* an accumulator with initial value zero to each arc of the network;
• **For each** example case from the training database, **do**:
Propagate the evidence from input nodes until the hypotheses layer;
For each arc reaching a hypothesis node, **do**:
If the reached hypothesis node corresponds to the correct class of the case
Then *backpropagate* from this node until input nodes, increasing the accumulator of each traversed arc by its evidential flow (Reward)
Else *backpropagate* from the hypothesis node until input nodes, decreasing the accumulator of each traversed arc by its evidential flow (Punishment).

After training, the value of accumulators associated to each arc arriving to the output layer will be between [-T, T], where T is the number of all cases present in the training set. The last step is the network pruning. It is performed by the following actions:

- remove all arcs whose accumulator is lower than a threshold (specified by a specialist);
- remove all neurons from the input and combinatorial layers that became disconnected from all hypotheses in the output layer; and
- make weights of the arcs arriving at the output layer equal to the value obtained by dividing the arc accumulators by the largest arc accumulator value in the network. After this pruning, the network becomes operational for classification tasks.

Despite its simplicity, CNM has many worthy features, as seen in the previous section. However, it has some drawbacks that limit its use, like: (a) in the initial phase, the generation of the network completely empty, representing all possible combinations for each hypothesis, is clearly unfeasible; (b) as recognized by the authors of the model [1] the full generation of all combinations of attribute-values may create many unreal hypotheses with respect to the majority of the applications; and (c) as a consequence of its knowledge representation form, CNM has its expressivity limited to Propositional Logic. Machado [1] suggests a criterion based on the "magic number" of Miller [9], seven plus or minus two, to establish the upper bound to the order of combinations. Feldens [2] proposes to control the growth of the combinatorial layer by building incrementally the network. The mechanism starts with a low combination order and increases the order to an upper one until an arbitrary limit. Our approach is addressed to this problem and may be seen as an alternative that can rise the order of attribute-values combinations generated from the example cases while keep control of the network growth.

3. PROPOSAL OF OPTIMIZATION

In the usual algorithm to build CNM, for each combination related to a hypothesis, a node is created in the intermediate layer. Also, there are only one-to-one relations between neurons of the intermediate and the hypotheses layers. In practice more than one hypothesis related to the same combination may exist (e.g., in production rule form: *If is_holiday and has_not_to_do Then go_to_the_movie* or *If is_holiday and has_not_to_do Then go_to_swim*). However, this fact leads to a high level of redundancies in the intermediate layer, where the AND nodes are located. Our proposal consists of transform the CNM architecture to a full-connected network. This transformation is possible by changing the relation between the AND nodes and the hypotheses layer. In Figure 2 this change is illustrated. For each hypothesis in the output layer, it is introduced the list of all combinations related to it, and so, an AND node can be referred to any number of output nodes.

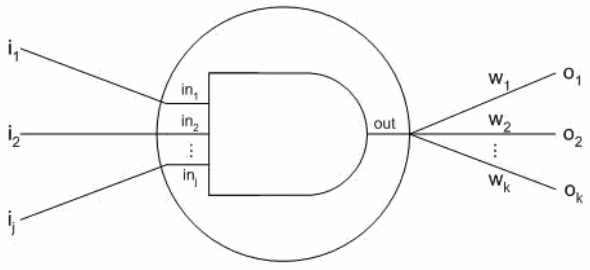


Figure 2 – The AND nodes shape

Considering Figure 1, let us suppose the following equivalences among attribute-value combinations C1=C5, C2=C6, C3=C7, and C4=C8. The resulting network with the proposed simplification is shown in Figure 3. It can be observed an expressive reduction in the network size, even in that simple example. It is expected a much bigger gain in real situations like the one presented in the fifth section that deals with dactyloscopy domain.

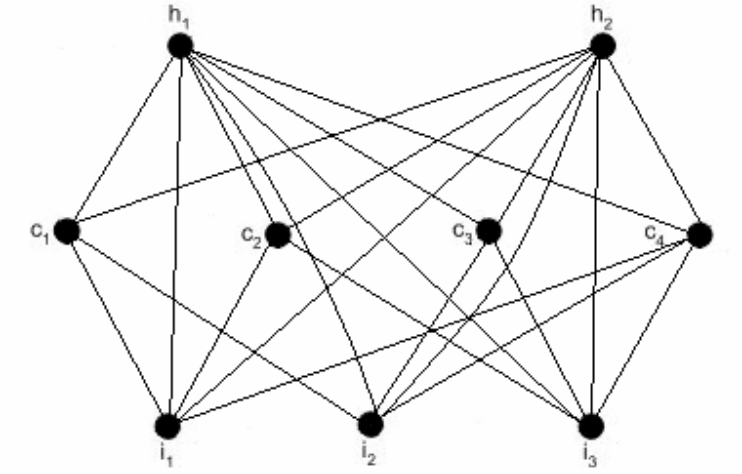


Figure 3 – Optimized architecture

4. THE DACTYLOSCOPY DOMAIN

The purpose of Biometrics is to identify an individual based on his or her physical characteristics. In conjunction with the resources offered by Information Technology, Biometrics offers interesting and effective solutions in the area of public safety, particularly in the identification of individuals involved in criminal activity. Dactyloscopy is a biometrics technique that has been widely used to identify criminals, given that it satisfies the requirements of the permanence, immutability, and singularity of fingerprints [8]. Dactyloscopy is the process by which individuals are identified through the examination of their fingerprints. The fingerprint is the mirror image of the digital pattern.

Dactyloscopic classification systems were developed to reduce the complexity of space and time required for the identification of fingerprints. Two major classification systems have been adopted around the world: Vucetich and Henry. The Brazilian police force employs the Vucetich system, the most widely used method in the world. The Vucetich system [8] defines four primary types of fingerprints with the following classifications: arches, internal loops, external loops, and whorls. Subsequently, the accidental, scar, and amputation types were added. These seven primary types are defined [7] according to the types below (see the primary types in Figure 4):

Figure 4 – Types of primary fingerprints

Arch - refers to the dactylogram made up of generally parallel and convex ridges that run or tend to run from one side of the print to the other and very often reveal angular or vertical ridges. The arch type is represented by number 1 or letter *A.*

Internal Loop: refers to the dactylogram that presents a delta to the observer's right and a nucleus composed of one or more ridges, which run from the left of the print toward the center, recurving and returning, or tending to return, to the

side from which they originated, thereby forming one or more loops. Loops involve the two-way movement of a papillary line, which must have perfect inflection. The internal loop type is represented by number 2 or letter *I*.
External Loop: refers to the dactylogram that reveals a delta to the observer's left and a nucleus composed of one or more ridges that run from the right of the print toward the center, recurving and returning, or tending to return, to the side from which they originated, thereby forming one or more loops. The external type is represented by number 3 or letter *E*.
Whorl: refers to the dactylogram characterized by the presence of a delta to the observer's left and right and a varied nucleus, which presents at least one curved ridge in front of each delta. The whorl type is represented by number 4 or letter W.
Accidental: refers to the dactylogram that does not fit in any of the four primary types cited above. It is represented by number 5.
Scar: refers to the dactylogram that presents a permanent mark caused by a cut, pustule, burn, or crushing, thereby making its classification within one of the 5 types cited above impossible and which is represented by number 6.
Amputation (or failure): refers to the type in which a total or partial loss of the phalange is evidenced, therefore compromising or even making impossible the classification of the primary type and which is represented by number 7.

If we create a fraction in which the numerator is the number formed by the algorisms that represent the pattern of the fingers of the right hand, extending from the thumb to the small finger, and the denominator constitutes the same number for the left hand, we arrive at what is known as *dactyloscopic formula*, or formula for short.

5. APPLICATION DESCRIPTION AND RESULTS

For the application developed it was used the data set described and pre-processed in [8]. In that work, the author had taken 855,000 records from "MECASinic" database, property of the Brazilian National Identification Institute. After discarding types 6 (scars) and 7 (amputations), 505,052 formulas remained that were segmented as follows:

- Subset A: formulas with a frequency near 1% or more (the formulas in this subset have more than of 5,000 observations each one). It corresponds to 67,364 observations and just 7 types of formulas;
- Subset B: 366,552 observations corresponding to formulas with frequencies between 0.0022% e 1% (the formulas in this subset have frequencies between 12 to 4,999 observations each one). There are 3.221 different formulas in this subset;
- Subset C: formulas with a frequency smaller than 0.0022% and less than 12 observations for each one. There are 68,136 observations and 34,906 different formulas in this subset;

Each record in the data sets contains one string of 10 positions representing the formula. One formula presents 10 values, one for each finger. For example, the string "3113431142" corresponds to a person with finger types (beginning with the right thumb) external loop, arch, arch, external loop, whorl, external loop, arch, arch, whorl, and internal loop. The classification task was defined as the prediction of the small finger. This task was chosen to illustrate one of the possible model applications, in this case to complement missing information of a dataset. Our experiments were carried out with the subsets A and B. No data mining techniques were applied to subset C because it has a sparse distribution of classes and does not follow any pattern, thus it is not statistically relevant. The identification of individuals involved in criminal activity, which have dactyloscopic formulae in the subset C were, handled ad-doc using a search algorithm because there were less than 12 individuals for each one formula.

Considering that our objective is to evaluate the effectiveness of our architecture in reducing the CNM intermediate layer growth, we adopted a simple scheme to build the model: 65% of each data set to the training phase and 35% to the test phase. Although it is a simple option, it is sufficient for our purpose.

The CNM intermediate sizes found is depicted in Table 1.

Table 1 – Comparing the proposed architecture with the usual model

Data set	Generated combinations (nodes in the intermediate layer)			Reduction rate (%)	Error rate (%)
	Usual model	Proposed architecture	Reduction		
A	99,156	63,762	35,394	35	18

B	439,641	224,303	21,5338	48	15

The results obtained show an important reduction in the growing rate of CNM that can strongly mitigate the combinatorial explosion of its intermediate layer, making the model viable to cope with bigger training sets.

Acknowledgements

We would like to thank Mr. Marcos Elias Cláudio de Araújo, a papilloscopist from the INI/DPF, who kindly provided the access to the "MECASinic" database.

References

[1] Machado, R. J, and Rocha, A. F., The combinatorial neural network: a connectionist model for knowledge based systems. In: Bouchon, B.; Yager, R. R.; Zadeh, L. A. *Uncertainty in Knowledge Bases*, Berlin, Germany: Springer Verlag, 1991, p.578-587.

[2] Feldens, M. A., *Engenharia da Descoberta de Conhecimento em Bases de Dados: Estudo e Aplicação na Área de Saúde*. Porto Alegre: CPGCC da UFRGS, Brazil, 1997. (M.Sc. Dissertation in Portuguese).

[3] Prado, H. A., Frigeri, S. R., Engel, P. M., A Parsimonious Generation of Combinatorial Neural Model. *Proc. of the IV Congreso Argentino de Ciencias de la Computación* (CACIC'98), Neuquén, Argentina, 1998.

[4] Beckenkamp, F. G., Feldens, M. A., Pree, W., Optimizations of the Combinatorial Neural Model. *Proc. of the 5th Brazilian Symposium on Neural Networks*. (SBRN'98), Belo Horizonte, Brazil, 1998.

[5] Prado, H. A. do; Machado, K.F.; Frigeri, S. R.; Engel, P. M. Accuracy Tuning in Combinatorial Neural Model. *Proc. of Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Bejing, China, 1999.

[6] Freeman, J. and Skapura, D. Adaptive Resonance Theory. In: *Neural Networks, Algorithms, Applications, and Program Techniques*. Reading: Addison-Wesley, 1992. 401p. p.292-339.

[7] INI - Instituto Nacional de Identificação, 1987. *Identificação Papiloscópica*, Departamento de Polícia Federal (DPF). Brasília. (Technical Report in Portuguese)

[8] Oliveira, M. G., *Otimização de Busca Decadactilar para Identificação de Impressões Digitais Utilizando Técnicas de Mineração de Dados*. CIC/UNB, Brasil, 2004. (M.Sc. Dissertation in Portuguese).

[9] Miller, G. A., The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 63, p.81-97, 1956.

Seguridad en ARAMCEL: Arquitectura basada en agentes móviles para Comercio Electrónico.

Sergio F. Castillo C.
Universidad Industrial de Santander, Escuela de Ingeniería de Sistemas,
Bucaramanga, Colombia,
scastill@uis.edu.co

Luis Antonio León Chacón
Universidad del Valle, Escuela de Ingeniería de Sistemas,
Tuluá, Colombia,
luisleonc@univalle.edu.co

Janeth Gissella Gómez Gualdrón
Universidad Industrial de Santander, Escuela de Ingeniería de Sistemas,
Bucaramanga, Colombia,
nanigiss@yahoo.fr

Abstract

Mobile agents are software entities that they can transport their program code and data from one computer to another through Internet. There are several security risks because as much the mobile agents as the servers with which interact are vulnerable to attacks and breaches of security. In this paper, three problems of security are focused: Authentication, Authorization and No Repudiation. In the context of the Architecture based on Mobile Agents for Electronic Commerce "ARAMCEL" a mechanism of security is presented which proposes a solution to those problems. The security model depends on a central server, reliable servers and the Infrastructure of public key (PKI). ARAMCEL's validation was carried out by means of the prototype ADAM: Application of the mobile agents to the e-commerce.

Keywords: Security, Electronic Commerce, Mobile Agent, Malicious Agent, Malicious Server.

Resumen

Los agentes móviles son entidades software que pueden transportar su código y datos desde un computador a otro a través de Internet. Hay varios problemas de seguridad porque tanto los agentes móviles como los servidores con que interactúan son vulnerables a ataques y brechas de seguridad. En este documento, se enfocan tres problemas de seguridad: Autenticación, Autorización y No Repudio. En el contexto de la Arquitectura basada en Agentes Móviles para Comercio Electrónico "ARAMCEL" se presenta un mecanismo de seguridad que propone una solución a esos problemas. El modelo de seguridad depende de un Servidor Central, servidores confiables y la Infraestructura de llave pública (PKI). La validación de ARAMCEL se realizó por medio del prototipo ADAM: Aplicación de los agentes móviles al comercio electrónico.

Palabras claves: Seguridad, Comercio Electrónico, Agente Móvil, Agente Hostil, Servidor Hostil.

1. Introducción

La aparición y crecimiento de Internet ha variado la forma de vivir y pensar de las personas en la sociedad; de igual forma este cambio ha afectado a las empresas, las cuales han modificado su concepción acerca de la manera de negociar e intercambiar información, bienes y servicios y han ingresado a la economía electrónica. Todo eso ha originado una nueva forma de negocios a través de Internet, llamada Comercio Electrónico (CE) [13].

Actualmente existen en el mercado algunos sistemas para CE, la mayor parte de ellos basados en el modelo Cliente / Servidor (C/S). Bajo este enfoque, cuando un cliente desea obtener un producto ó servicio debe interactuar con el sitio del proveedor, enviando peticiones y recibiendo respuestas, tantas veces como sea necesario para cumplir su tarea, esto incrementa el tráfico en la red, el tiempo invertido para realizar la actividad y los costos de conexión (Figura 1).

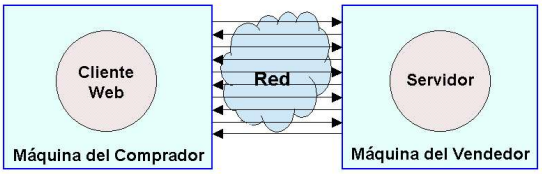


Figura 1. Interacción basada en el modelo C/S.

Buscando la forma de disminuir estos inconvenientes, se ha propuesto el modelo de Agentes Móviles (AM) [7], el cual por sus características disminuye el tráfico en la red, supera los inconvenientes de una conexión intermitente y ofrece soluciones para la poca disponibilidad de tiempo en los usuarios. En la Figura 2 se presenta el tráfico generado en la red por la interacción necesaria para realizar tareas que utilizan recursos que se encuentran ubicados en un equipo remoto, utilizando el modelo de AM.

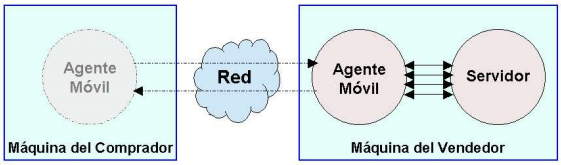


Figura 2. Interacción basada en el modelo de AM.

Para realizar esas tareas un agente móvil cuenta con características que lo hacen autónomo, reactivo y proactivo [2]; además el agente puede representar un usuario dentro de una transacción comercial, puesto que en ellas se requiere la intervención de entidades que negocien o actúen con base en exigencias, necesidades, intenciones, gustos, etc.; en algunos casos la realización de transacciones requiere de acceso en tiempo real a recursos que se hallan distribuidos en la red, es de esta forma que la movilidad de los agentes es una ventaja para realizar aportes en este entorno.

Pero el modelo de agentes móviles también tiene desventajas y se deben a la seguridad [5]. Los agentes móviles se ejecutan en servidores remotos y allí interactúan con otros agentes en la red, por tanto es inevitable que tanto los agentes como los servidores estén sujetos a ataques de entidades hostiles. Estas entidades incluyen agentes hostiles, servidores hostiles e intrusos intentando atacar agentes móviles o servidores, por esa razón el componente de seguridad de ARAMCEL es un aspecto crucial a tener en cuenta durante el desarrollo de sistemas basados en agentes móviles.

Por las razones mencionadas anteriormente surge la arquitectura ARAMCEL, que pretende apoyarse en el modelo de agentes móviles para dar una solución a esas necesidades. ARAMCEL ha sido propuesta como Tesis de Maestría en Informática en la Universidad Industrial de Santander y su objetivo general es considerar los principales problemas relacionados con la seguridad requerida en un contexto de Comercio Electrónico y así permitir el desarrollo de aplicaciones en ese campo. Para comprender ARAMCEL, en las siguientes secciones se hace una descripción de los componentes, funcionamiento, amenazas de seguridad, así como el modelo de seguridad y la solución propuesta a las amenazas mencionadas.

1. ARAMCEL: Arquitectura basada en el modelo de Agentes Móviles para Comercio Electrónico.

ARAMCEL (Figura 3) es una arquitectura que permite realizar transacciones de Comercio Electrónico utilizando el modelo de Agentes Móviles. La especificación de ARAMCEL está basada en el lenguaje de programación JAVA, en los conceptos del modelo de AM y en las necesidades presentes en un entorno de Comercio Electrónico.

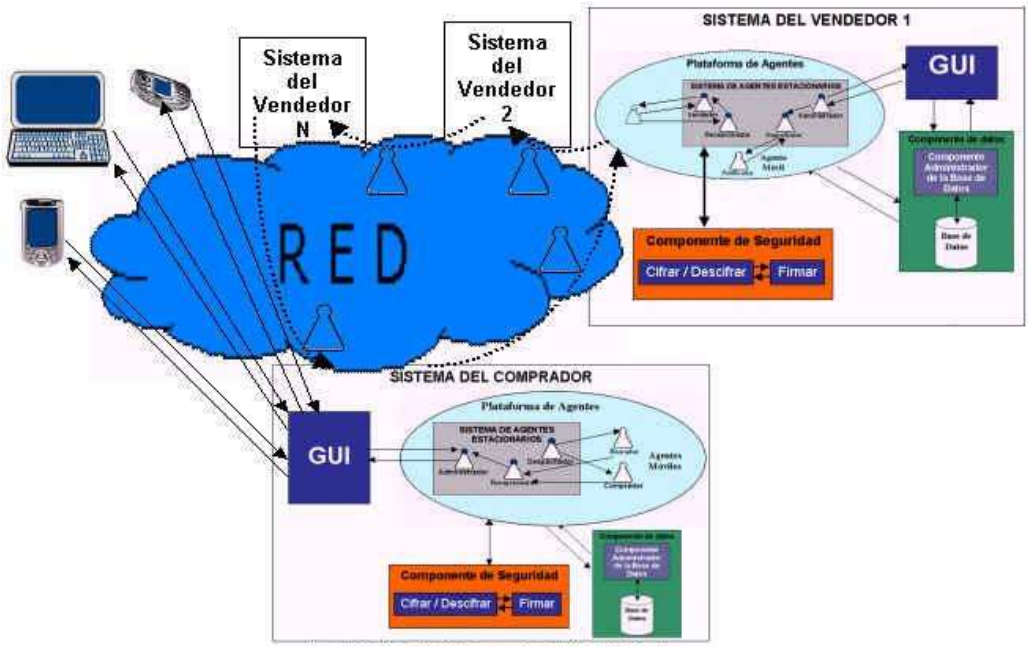


Figura 3. Arquitectura ARAMCEL.

1.1. Descripción de ARAMCEL.

En la Arquitectura ARAMCEL se utiliza el modelo de negocios en el cual el Comprador, con base en sus requerimientos, envía agentes móviles que acceden a los productos ofrecidos por el Vendedor. ARAMCEL está distribuida en dos sistemas definidos, el Sistema del Comprador y el del Vendedor.

1.1.1. Sistema del Comprador

El sistema del Comprador está conformado por la interfaz gráfica de usuario, la plataforma de agentes, el componente de seguridad y el componente de datos (Figura 4).

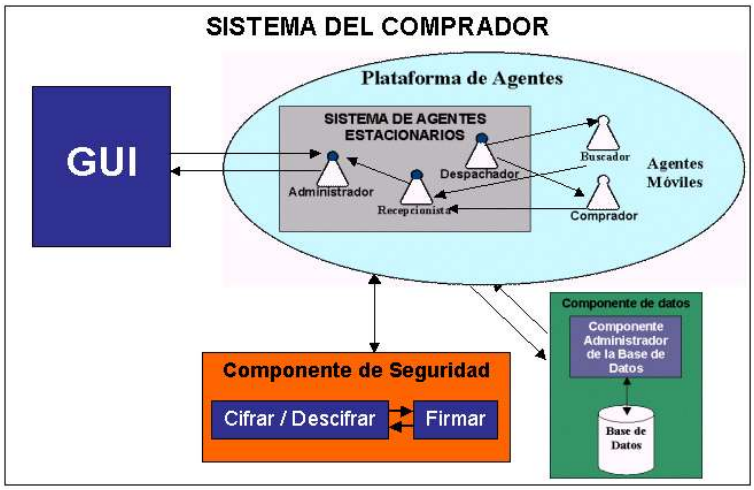


Figura 4. ARAMCEL - Sistema del Comprador.

El sistema del Comprador se encuentra instalado en un equipo denominado Servidor de Compradores (SC), al cual acceden los usuarios Compradores por medio de la interfaz de usuario. Un Comprador puede acceder al sistema desde cualquier dispositivo que posea conexión a la red (computador, asistente digital personal, celular, computador portátil, etc.). Los usuarios previamente registrados, sólo necesitan un navegador para Internet, mediante el cual se conectan al SC y pueden usar el sistema.

A continuación se hace una descripción de cada uno de ellos.

- *Interfaz Gráfica de Usuario (GUI):* Permite que el usuario ingrese información al sistema; esta información corresponde a requerimientos, preferencias sobre artículos y demás datos que sean necesarios. Además se utiliza como medio para mostrar al usuario la información que los agentes han encontrado.

- *Plataforma de Agentes*: En ella se encuentra el Sistema de Agentes Estacionarios y los agentes móviles. Los agentes estacionarios son aquellos que permanecen en la máquina del Comprador y cumplen funciones como el despacho de agentes a los sitios de los Vendedores y la recepción de los agentes móviles cuando retornan con la información que han recolectado. Los Agentes Móviles son agentes creados para la búsqueda y compra de artículos, llevan la información de las preferencias del usuario sobre los artículos que desea y el itinerario con los sitios a visitar.

- Componente de Seguridad: Este componente está subdividido en dos módulos, el primero se encarga del cifrado y descifrado de los datos y el segundo de los procesos de firmado y verificación de firmas. Este componente es utilizado por el Agente Despachador para solicitar el cifrado y firmado de la información que va a transportar el agente móvil. También es usado por el Agente Estacionario Recepcionista del Comprador para verificar la firma del Vendedor y descifrar la información que se recibe del agente móvil.

- *Componente de Datos:* Este componente está conformado por la Base de Datos y el Componente Administrador de la Base de Datos. La primera contiene la información relacionada con las solicitudes del Comprador y el resultado de los procesos realizados por el Agente Móvil; también se almacena una lista con las direcciones de los sitios de los Vendedores y el tipo de artículos que ofrecen, ese listado es usado para la generación del itinerario del agente móvil. El Componente Administrador de la Base de Datos es el encargado de realizar las consultas y modificaciones a la información almacenada en ella.

1.1.2. Sistema del Vendedor

El sistema del Vendedor contiene la interfaz gráfica de usuario, la plataforma de agentes, el componente de seguridad y el componente de datos (Figura 5). El nombre de los componentes es el mismo que en el sistema del Comprador, pero cambia la funcionalidad en algunos de ellos como se describe a continuación.

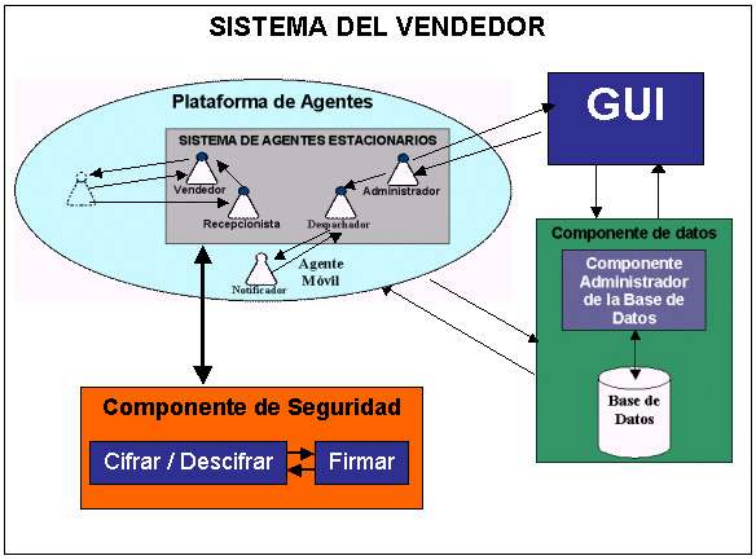


Figura 5. ARAMCEL - Sistema del Vendedor

- *Interfaz Gráfica de Usuario (GUI):* Permite al usuario (dueño o administrador del sitio del Vendedor) realizar modificaciones en los datos existente en ese sitio y obtener la información de las transacciones comerciales realizadas.

- *Plataforma de Agentes:* Entre los agentes estacionarios de esta plataforma, se encuentra el agente Recepcionista, encargado de asignar un agente Vendedor que interactúe con el agente móvil que hace una consulta o solicita una compra. El agente Despachador es el que crea y envía al Agente Notificador, éste es un agente móvil del Vendedor que se encarga de informar a los Compradores sobre las ofertas vigentes en el sitio, nuevas adquisiciones y demás mensajes que desee enviar el Administrador de la tienda.

- *Componente de Seguridad:* Al igual que en el sistema del Comprador, está subdividido en dos módulos, el primero se encarga del cifrado y descifrado de los datos y el segundo de los procesos de firmado y verificación de firmas. Este componente es utilizado por el Agente Vendedor para el procesamiento de la información recibida del agente móvil del Comprador y la que a su vez va a ser entregada dentro del proceso de venta; además tiene la función de revisar la integridad de los agentes.

- *Componente de Datos:* Esta conformado por la base de datos que contiene la información referente a los artículos que ofrecen en ese sitio, ventas realizadas y ofertas; y por el Administrador de la base de datos que permite realizar consultas y modificaciones a la información almacenada en ella.

1.2. Funcionamiento de ARAMCEL

Para realizar las transacciones de Comercio Electrónico, ARAMCEL representa los componentes y la interacción entre ellos, donde un Usuario Comprador puede enviar un agente para que lo represente e interactué con otros agentes en sitios remotos de la red y así obtener el artículo que busca. Los siguientes numerales dan una breve descripción del funcionamiento de ARAMCEL.

Sistema del Comprador.

Para ingresar al Sistema del Comprador, el usuario se conecta a la GUI utilizando un protocolo de comunicación segura y solicita el inicio de sesión; para esto debe ingresar su nombre de usuario "*login*" y la clave de acceso "*password*".

Una vez que sea válida la entrada, el Usuario puede hacer búsquedas en la red de los artículos que quiere. El sistema mediante sus agentes móviles toma la información de los artículos y las preferencias del Usuario, como puede ser el tiempo de entrega del artículo, costo, cantidades de unidades, etc.; cuando se tienen todos los datos, los agentes se desplazan a través de la red a cumplir con la búsqueda de información y cuando la recolectan, regresan a su sitio de origen a reportar los resultados obtenidos.

El Usuario Comprador puede finalizar su sesión en el sistema y después reconectarse, para revisar el listado que contiene los resultados que trajeron los agentes; con base en esa información, puede dar la orden de crear un agente móvil Comprador que viaje al sitio elegido y procese la orden de compra.

Sistema del Vendedor.

En el caso de los sitios correspondientes a cada Vendedor, los agentes estacionarios son los encargados de interactuar con los agentes móviles buscadores de información y con aquellos que llegan a realizar procesos de compra.

La GUI de este sitio permite al propietario o administrador verificar los artículos existentes, obtener información de ventas realizadas, crear ofertas, actualizar precios, modificar la cantidad de artículos cuando llega nuevo inventario y realizar promociones.

1.3. Transacciones básicas en ARAMCEL

Las transacciones corresponden a los procesos de Comercio Electrónico que se realizan en ARAMCEL, como son la búsqueda y compra de artículos. A continuación se describen de manera general los pasos presentes en cada una de ellas.

Búsqueda de información.

- El Usuario Comprador (UC) hace una solicitud de artículos a través de la Interfaz Gráfica de Usuario (GUI).
- El Agente Estacionario Administrador del Comprador (AEAC) procesa la petición y le entrega los datos de los artículos y las preferencias suministradas por los usuarios al Agente Estacionario Despachador del Comprador (AEDC).
- El AEDC con esos datos crea un Agente Móvil Buscador (AMB) por cada artículo de la lista, con las respectivas preferencias.
- El AMB solicita al AEDC información de sitios de Vendedores que ofrezcan el tipo de artículo que necesita. Esos sitios son evaluados por medio de un puntaje que los clasifica con base en la calidad del servicio prestado.
- El AMB viaja siguiendo un itinerario que ha formado con base en los sitios recibidos.
- Al llegar al sitio del Vendedor el AMB establece comunicación con el Agente Estacionario Recepcionista del Vendedor (AERV).
- El AERV le asigna un Agente Estacionario Vendedor (AEV) para que interactúe con el AMB.
- El AMB solicita el artículo al AEV y espera la cotización que incluye precio, cantidad y tiempo de entrega del artículo.
- El AMB almacena la oferta recibida, si cumple con las preferencias del UC, y continua su recorrido.
- Una vez que haya finalizado el itinerario, el AMB retorna a su sitio de origen y reporta al AERC los resultados obtenidos en su viaje.

Compra de Artículos

- El UC a través de la GUI revisa los resultados de las búsquedas de artículos y decide cuales comprar.
- El AEAC toma la información de los artículos que desea comprar el UC y la envía al AEDC.

- El AEDC crea un Agente Móvil Comprador (AMC) por cada artículo.
- Cada AMC viaja al sitio donde se encuentra el artículo seleccionado por el UC.
- El AMC llega y el AERV le asigna un AEV que recibe la solicitud del artículo y entrega los resultados.
- El AMC envía el código de pago y solicita el código de aceptación al AEV.
- El AEV entrega el código de aceptación al AMC, quien regresa al origen a entregar los resultados al AERC.

Estas transacciones corresponden a un escenario sin inconvenientes de seguridad. En la siguiente sección se presentan las amenazas de seguridad que se pueden manifestar al realizar una transacción de búsqueda o una compra en sitios remotos.

2. Amenazas de seguridad en ARAMCEL

La movilidad de agentes y la interacción con otras entidades en la red, hace surgir unos requerimientos de seguridad en ARAMCEL que pueden ser clasificados en dos categorías. La primera de ellas se refiere a los problemas de seguridad que se presentan en el modelo de los Agentes Móviles y la segunda los aspectos que se requieren para garantizar la seguridad en un entorno de Comercio Electrónico. Los siguientes numerales detallan cada una de las categorías mencionadas.

2.1. Amenazas de Seguridad Inherentes al Modelo de Agentes Móviles.

La seguridad ha sido tradicionalmente un inconveniente en los sistemas informáticos, pero el modelo de agentes móviles por sus características presenta un nuevo tipo de inconvenientes de seguridad. Los dos problemas principales son [8, 12, 13]:

El Problema del Agente hostil.

Cuando los agentes móviles de ARAMCEL llegan a cada sitio de su itinerario, necesitan algunos permisos de usuario para poder ejecutar su código. Tradicionalmente el control de acceso a los equipos de una red se hace por medio de contraseñas, en donde cada usuario se autentica al inicio con su *login* y su *password*; pero este esquema funciona para sistemas estáticos y no para agentes móviles, puesto que el agente debe llevar una contraseña para cada sitio de su itinerario, lo que aumenta su tamaño y así mismo incrementa los requerimientos de seguridad para poder proteger esas contraseñas.

Entre los agentes puede existir agentes hostiles que realizan acciones perjudiciales, como el acceso no autorizado, alteración de los recursos locales (datos y llamadas al sistema), sobrecarga de esos recursos o incluso daños a la integridad del servidor. En la figura 6 se presenta un agente hostil que llega a un sitio y ha sido diseñado para realizar labores que atentan contra la integridad del servidor, como es la utilización no autorizada de los recursos para obtener información privada.

Figura 6. Agente Hostil.

El problema del agente hostil ha generado tres tipos de amenazas a los servidores, las cuales se detallan a continuación:

- *El acceso no autorizado a los recursos.* Cuando un agente móvil llega a cada sitio de su itinerario, ejecuta su código usando los recursos del computador del Vendedor. Por esta razón, los recursos de la máquina receptora están expuestos al ataque de un agente hostil, puesto que si no existen los mecanismos de control apropiados, ese agente podría acceder a información privada del sistema. En el caso del Comercio Electrónico, la información que almacenan los Vendedores está relacionada con sus datos comerciales, de esa forma, si un agente accede a ella puede alterarla y tomar ventaja de su contenido para fines ilegales.

- *Sobrecarga de los recursos del servidor.* Es necesario proveer medidas de seguridad adecuadas para prevenir el consumo excesivo de recursos por parte de agentes hostiles, puesto que al sobrecargar el servidor hace que el tiempo de respuesta se incremente o que no haya respuesta a los agentes de los demás Compradores que buscan acceder a la información. En el caso particular de ARAMCEL los sitios de los vendedores son clasificadas con

base en su servicio mediante un puntaje, por lo tanto el problema mencionado podría ocasionar la reducción de esa valoración.

- *Creación de agentes residentes en el sitio del Vendedor.* Otro de los ataques de los cuales puede ser víctima un servidor por parte de un agente hostil, es la creación de otros agentes que residan en el servidor. Con esos nuevos agentes puede suceder cualquiera de los dos ataques anteriormente mencionados, ó que se creen tantos agentes que también sobrecarguen los recursos. En la mayor parte de los sistemas de Comercio Electrónico actuales se utiliza un agente "esclavo", creado por el agente móvil, para que resida en el sitio del Vendedor y asegure la reserva de un producto; este proceso no se realiza de la misma forma en ARAMCEL, en secciones siguientes se explicará el proceso utilizado.

La solución tradicional al problema del agente hostil se realiza por medio de restricción de privilegios a los agentes, pero eso puede ocasionar que los agentes sean incapaces de lograr sus metas, por lo tanto es necesario buscar mecanismos alternos que provean seguridad y permitan la realización de las transacciones.

El problema del servidor hostil

Un servidor hostil es una máquina que intenta espiar ó manipular el código, los datos o el control de flujo del agente, proporciona llamadas falsas al sistema, ejecuta código del agente incorrectamente, invierte la ingeniería y manipula su código ó los secretos comerciales; puede hacerlo mediante la clonación del agente, proporcionando información errónea ó interceptando el agente. Un ejemplo del ataque de un servidor hostil a un agente móvil puede observarse en la figura 7.

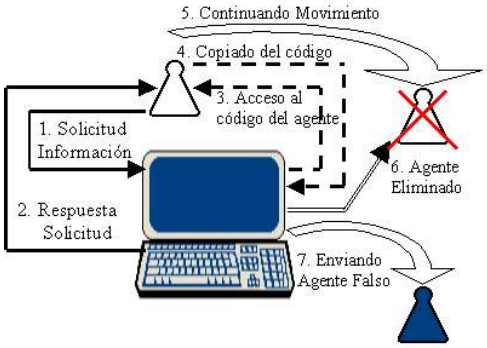


Figura 7. Servidor Hostil.

Cuando un agente móvil llega a un determinado sitio, ya no depende de su máquina de origen, por el contrario, la máquina receptora debe instanciar la clase que dio origen al agente, leer el estado de datos que se encuentra serializado y ejecutarlo dentro de ella, por lo cual el código y datos del agente móvil quedan totalmente expuestos. Los ataques de Servidores contra agentes son de diversos tipos y pueden resumirse como sigue:

- *Clonación del Agente*. Uno de los ataques al agente móvil consiste en que un servidor captura y construye una réplica de él, de esa manera, se hace pasar por el agente original pero para los propósitos que ese servidor programa. En el caso del Comercio Electrónico un Vendedor puede clonar un agente para enviarlo a buscar información privada y de esa forma acceder a los secretos comerciales de sus competidores.

- *Negación de servicio.* Los ataques de negación del servicio pueden ser de dos tipos, el primero ocurre cuando el servidor impide que el agente continúe su itinerario, el segundo consiste en permitir al agente continuar, pero no al sitio que tiene determinado en el itinerario, sino a aquel que el servidor elige. Este segundo tipo de ataque es frecuente en los sistemas de Comercio Electrónico, puesto que evita que el agente vaya al sitio del competidor y puede enviarlo al de un socio ó a otra sucursal del mismo Vendedor.

- *El ataque del hombre en el medio*. Un agente móvil puede ser interceptado y modificado mientras se desplaza de un servidor a otro. La forma de realizarlo es utilizando un programa denominado "monitor de comunicación" que se encarga de observar los mensajes intercambiados por los sistemas de agentes para extraer o modificar la información o el código de los agentes. Con este tipo de ataque un Vendedor hostil puede espiar o alterar los datos que los demás Vendedores han entregado al agente.

- *Modificación de los datos que lleva el agente*. Cuando un servidor ejecuta un agente móvil puede observar su contenido, por esta razón, si no se establecen los mecanismos de seguridad necesarios, un Vendedor hostil puede modificar los datos que le han sido entregados por otros Vendedores al agente y de esa forma competir deslealmente.

2.2. Principales problemas del Comercio Electrónico

Con el auge de Internet y las comunicaciones aparecen avances en el desarrollo tecnológico que buscan aumentar la competitividad empresarial; en el caso del Comercio Electrónico, el modelo de agentes móviles ofrece grandes oportunidades para realizar transacciones. Así mismo se encuentran en el mercado sistemas basados en agentes móviles, se cuenta con estándares, plataformas y productos que facilitan su desarrollo, pero aún el Comercio Electrónico no ha llegado al auge esperado; una de las razones para que esto suceda son los problemas de seguridad existentes [1]. Por eso se hace necesario buscar soluciones para minimizar el riesgo al realizar transacciones de Comercio Electrónico y la manera de lograrlo es implementar los siguientes aspectos de seguridad [4]:

Autenticación.
Asegura que el escritor de un documento o el remitente de un mensaje es quién dice ser. Para lograr esta autenticación están surgiendo métodos de gran importancia que conducen a un manejo efectivo de la seguridad y la confidencialidad. Algunos de los métodos utilizados son: la Identificación Biométrica, los Documentos de Identificación y la Información Confidencial.

En la autenticación mediante información confidencial se utiliza una combinación de caracteres como contraseña, donde sólo el usuario y el proveedor de ella pueden conocerla o se utiliza una firma digital. Una firma digital es una identificación electrónica, creada por medio de un computador para representar a la persona o entidad, con el mismo efecto legal que tendría una firma hecha a mano.

Autorización.
Asegura que el usuario que va acceder a la información posee los permisos de seguridad necesarios, por tanto la información en la red permanece privada. Se puede cumplir usando métodos matemáticos complejos para realizar procesos como el de cifrado y descifrado.

No-repudio.
Asegura que ninguno de los implicados en la transacción puede negar su participación en ella, así mismo, los participantes no podrán rechazar las obligaciones contractuales adquiridas en el negocio; el creador del mensaje no puede negar que lo envió, y el receptor del mensaje no puede negar que lo recibió. Este aspecto es útil principalmente por razones comerciales y legales. Hay dos clases de No-Repudio:

- *De origen*: Provee al receptor del mensaje con la prueba de su origen, la cual podrá emplearse para defenderse de la negación del emisor sobre el envío del mensaje. Actualmente se está empleando la firma digital para evitar el no-repudio del emisor, la firma es anexada al mensaje enviado.

- *De recepción*: Provee al emisor del mensaje con la prueba de su entrega, la cual podrá ser empleada para defenderse de la negación del receptor. Para evitar el no-repudio del receptor actualmente se está optando por incluir una entidad, es decir, una tercera parte confiable, que en la mayoría de los casos es una autoridad de certificación, que actúa como "notario" dentro de la transacción y verifica la recepción del mensaje.

Confidencialidad.
Asegura que ninguna entidad ajena a la transacción pueda acceder a los datos que utilizan, de la misma manera los actores (personas, programas, procesos, etc.) de esa transacción tan sólo conocerán aquella información necesaria para su realización.

Integridad.
Asegura que un mensaje no ha sido modificado mientras es transportado. En Comercio Electrónico, el contenido de una transacción no debe ser alterado por alguna de las partes implicadas en ella, ni por una tercera parte no involucrada inicialmente; así mismo, la transacción debe poder ser perfectamente reconstruible frente a terceros en caso de disputas legales. Una forma para lograrlo es mediante el uso de firmas digitales con criptografía de llave pública, en las cuales es necesario registrar la llave pública antes de la transacción.

3. El Modelo de Seguridad en ARAMCEL

Dada la amplitud del problema de seguridad, en ARAMCEL se seleccionaron los aspectos de seguridad autenticación, autorización y no-repudio como los más relevantes en esta investigación. Esos problemas de seguridad se pueden resolver con la ayuda de técnicas para el cifrado de información. Hay dos esquemas para realizar ese cifrado:

- *Esquema común*: En este esquema la misma llave es guardada secretamente, se usa para el cifrado y descifrado de la información, se le llama cifrado simétrico. En este caso el autor del mensaje debe utilizar la clave para cifrarlo y el destinatario debe poseer esa misma clave para poder descifrarlo.

- *Esquema de llave pública*: En este esquema se usan llaves diferentes para el cifrado y descifrado, se denomina cifrado asimétrico, una llave se guarda confidencialmente y la otra es pública. El término utilizado para referirse a la infraestructura de seguridad creada con base en criptografía de clave o llave pública es PKI (Public Key Infrastructure) [6, 11]. PKI también permite la gestión de certificados digitales, donde un certificado es un documento firmado digitalmente por una persona o entidad confiable denominada Autoridad de Certificación (CA), que enlaza cierta información perteneciente a un sujeto con su llave pública para poder asegurar que la entidad con que se está tratando es quién dice ser.

En ARAMCEL, para encontrar una solución se utiliza la infraestructura de llave pública (PKI); PKI tiene la ventaja de no tener que compartir información confidencial entre las entidades involucradas en el intercambio de información, además permite realizar la autenticación de mensajes puesto que si una llave se utiliza para cifrar un mensaje sólo se puede descifrar con la otra.

Es importante aclarar que dentro de las políticas de ARAMCEL se tiene que los Vendedores y Compradores deben estar registrados para poder acceder al sistema, de esta forma un componente Administrador dentro del Servidor Central actúa como entidad de certificación expidiendo las llaves correspondientes y entregando la información que tanto Vendedores como Compradores deben tener. Cada sitio en ARAMCEL posee una llave privada y una llave pública, así mismo tiene acceso a las llaves públicas de los demás sitios, de esa manera los datos de respaldo generados durante la transacción son cifrados y firmados por el emisor.

Por ejemplo, el Comprador A posee una llave pública CP y una llave privada CR, el sitio del Vendedor N posee una llave pública VP y una llave privada VR. De esta forma, si un agente es enviado a realizar una transacción llevará cifrada la información que debe entregar al Vendedor, con la llave pública del Vendedor, por lo tanto, únicamente la llave privada correspondiente puede ser usada para leer la información cifrada por esa llave pública, como se muestra en la figura 8. La utilización de estas llaves ayudan a autenticar al creador de un mensaje.

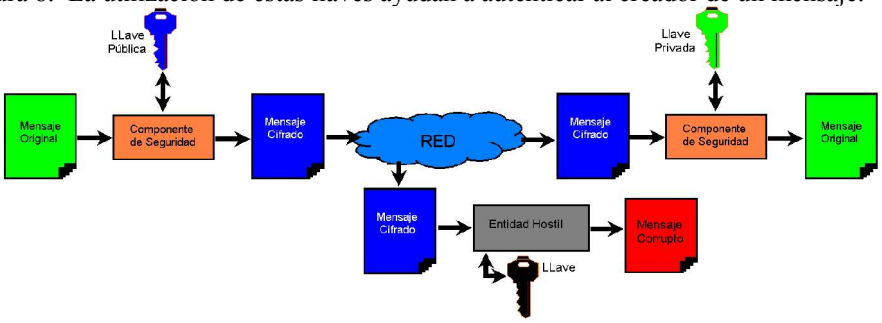


Figura 8: Esquema de Cifrado / Descifrado.

3.1. Autenticación

El esquema de autenticación de ARAMCEL se basa fundamentalmente en la firma digital. Cuando un Comprador envía un agente a realizar un proceso de búsqueda, este lleva consigo el identificador del usuario, cifrado con la llave privada del Comprador. Este esquema proporciona gran versatilidad y resuelve el problema de autenticación, además presenta una gran ventaja al utilizarlo para autentificar los agentes móviles, ya que cuenta con la certeza que la información no ha sido alterada y se sabe con seguridad quién los envía. Para realizar la autenticación, los agentes móviles de ARAMCEL han sido divididos en cuatro partes como se muestra en la Figura 9, donde cada una de ellas está diseñada para apoyar el protocolo de seguridad. A continuación se presenta su descripción:

- La identidad del sitio del comprador (C_Id) que lo creó.
- Una cabecera que contiene el identificador del agente (A_Id) y el resultado de una función HASH [9, 10] aplicada al código. Esos valores están cifrados con la llave privada del Comprador, de tal manera que sólo utilizando la llave pública de ese Comprador pueden ser descifrados.
- El código del agente puede verificarse mediante una función HASH, el resultado de ella se compara con la que lleva cifrada en la cabecera para indicar que ese agente proviene de un sitio confiable y no ha sido modificado.
- Los datos cifrados que el agente recoge de los diferentes Vendedores a lo largo de su itinerario.

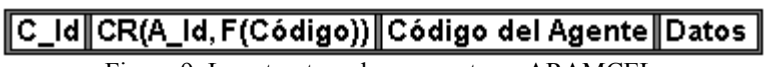


Figura 9: La estructura de un agente en ARAMCEL.

El uso de llaves y certificados es crucial para la autenticación, por ello ARAMCEL fortalece la seguridad con base en los conceptos de la clase de Java llamada *Keystore* [10] que representa una colección en memoria de llaves y certificados. Esa clase maneja dos tipos de entradas:

- *Llave:* este tipo de entrada del keystore maneja una llave confidencial de cifrado, la cual es almacenada en un formato protegido mediante una contraseña para evitar el acceso no autorizado. Generalmente, una llave almacenada en este tipo de entrada es una llave secreta o una llave privada acompañada por un certificado que corresponde a la llave pública. Las llaves privadas y los certificados son usados para que una entidad se auto autentique.

- *Certificado confiable:* este tipo de entrada contiene un certificado con una llave pública perteneciente a otra entidad. Es llamado un certificado confiable porque el propietario del keystore asegura que la llave pública dentro de ese certificado realmente pertenece a la entidad identificada en el certificado como el propietario de la llave.

Por seguridad se tiene una contraseña para todo el contenido y además cada entrada en el keystore se protege por una contraseña diferente.

3.2. Autorización

Los Agentes Móviles que llegan a cada sitio se consideran no confiables y dentro del archivo de políticas de seguridad se establece que los agentes sólo tienen acceso al envío de mensajes; de esa forma se asignan permisos que restringen a los agentes evitando acciones hostiles pero con la capacidad suficiente para interactuar con el sistema. Es necesario ser un usuario Comprador o Vendedor autorizado por el sistema con sus respectivas llaves públicas y privadas para cifrar o descifrar información; sólo quién esté autorizado, puede realizar el proceso de descifrado de la cabecera del agente y reconocer si ese agente pertenece a un usuario. Una vez el agente ha sido autenticado, se le autoriza para que use los recursos establecidos para él, con base en las políticas de seguridad del sistema.

Para el proceso de compra se requiere la previa autenticación tanto del agente Vendedor como del agente Comprador; el código de pago que lleva el agente Comprador va cifrado con la llave pública del Vendedor, para que sólo el Vendedor autorizado pueda usarlo; el código de aceptación de la transacción, va cifrado con la llave pública del Comprador para evitar que usuarios compradores no autorizados usen el código.

3.3. No-Repudio

En ARAMCEL, se utilizan las firmas digitales para evitar el repudio de origen y recepción. Cuando el agente móvil entrega el identificador de su usuario y el código de pago cifrado y firmado, está entregando la prueba que acredita que el Comprador envió el agente y ordenó la compra, de esa forma se evita el repudio de origen. Por su parte, cuando el Agente Vendedor, entrega el código de aceptación de la transacción, cifrado y firmado, se puede verificar que el Vendedor aceptó la transacción y por tanto se evita el repudio de recepción.

3.4. Solución al problema del Agente hostil

El lenguaje de programación Java [10] provee funcionalidades que permiten implementar políticas de seguridad basadas en un mecanismo de autenticación y autorización; en ARAMCEL, los permisos de archivos restringen al agente el uso de recursos en el computador del Vendedor, al igual que las operaciones que esos agentes pueden realizar. Además en ARAMCEL el sistema del Comprador que crea el agente, le añade una cabecera que contiene el identificador del propietario y cifrados con la llave privada del Comprador, el identificador del agente y el resultado de una función aplicada al código. Cuando un agente llega al sitio del Vendedor, la identidad del Comprador se verifica al aplicarle la llave pública del Comprador. El descifrado autentica al Comprador y revela la identidad del agente; en caso de no realizarse el descifrado de la cabecera del agente Comprador o si el agente intenta exceder su límite de permisos mientras se ejecuta, el Vendedor aborta la ejecución del agente y reporta al Servidor Central el ataque.

El Servidor usa las características de seguridad para definir los privilegios apropiados y restringir el acceso del agente a los recursos del sistema. Además ARAMCEL no permite la creación de agentes residentes, puesto que para reservar un artículo es suficiente con la cotización entregada y cifrada con la llave privada del Vendedor y nuevamente cifrada con la llave pública del Comprador. Los Servidores son protegidos de los agentes hostiles, pero los agentes aún están expuestos a ataques; utilizando las opciones de seguridad, un agente solicita cierto nivel de protección, pero esa solicitud depende del servidor en el cual está ejecutándose. Por tanto, no es posible para los agentes, protegerse de servidores hostiles que ignoran los parámetros de protección pedidos por ellos; se requieren técnicas adicionales para proteger a los agentes de esos servidores hostiles.

3.5. Solución al problema del Servidor hostil

Los agentes Compradores que llegan al sitio de un Vendedor ejecutan su código dentro de esa máquina, exponiéndose al problema del servidor hostil. En ARAMCEL, cada sitio de un Vendedor se encuentra en una lista de servidores confiables que están dentro del SC y han sido previamente acreditados por el sistema. Además, con la ayuda de PKI las entidades certificadas por el Servidor Central tendrán el par de claves necesarias para el cifrado y descifrado de mensajes y datos con los agentes. En caso que un servidor trate de modificar los datos o el código del

agente, se verifica por medio de la información contenida en la cabecera del agente; en ese caso el agente es tratado como hostil y eliminado por el sistema donde se encuentre. Si un agente desaparece cuando recorre el itinerario, ARAMCEL permite el envío de un agente móvil Fiscalizador que sigue el mismo recorrido y envía mensajes cuando llega y sale de cada servidor, para informar al sistema el sitio donde ocurrió el ataque.

Al encontrar un sitio hostil inmediatamente el Servidor Central lo marcará como No Confiable y será eliminado de la lista de sitios posibles para viajar, en caso de no encontrarlo los agentes móviles que tengan que pasar por esos sitios serán acompañados por agentes fiscales, durante la cantidad de viajes que el Administrador del Servidor Central con base en su autonomía considere necesario.

3.6. Protocolo de Seguridad

Los aspectos más relevantes del protocolo se presentan en la figura 10 y su proceso, en forma general, se describe a continuación.

Cuando el Agente Móvil del Comprador (AMC) llega al dominio del Vendedor solicita la asignación de un agente vendedor y recursos en el servidor. Antes de cumplir con la solicitud, el Agente Estacionario Recepcionista del vendedor (AERV) lee los datos del agente y se los envía al Componente de Seguridad, para que realice la autenticación del agente y del Usuario Comprador. El Componente de Seguridad busca el certificado digital del comprador y extrae la Clave Pública, confirma la existencia del usuario y revisa la integridad del código del agente; descifra el A_Id y la F(código), aplica la función Hash al código del agente y compara la función obtenida con la descifrada. Cuando el AMC es autenticado, se le asignan recursos y un Agente Estacionario Vendedor (AEV); con este último, el AMC realiza la interacción del proceso de Compra / Venta.

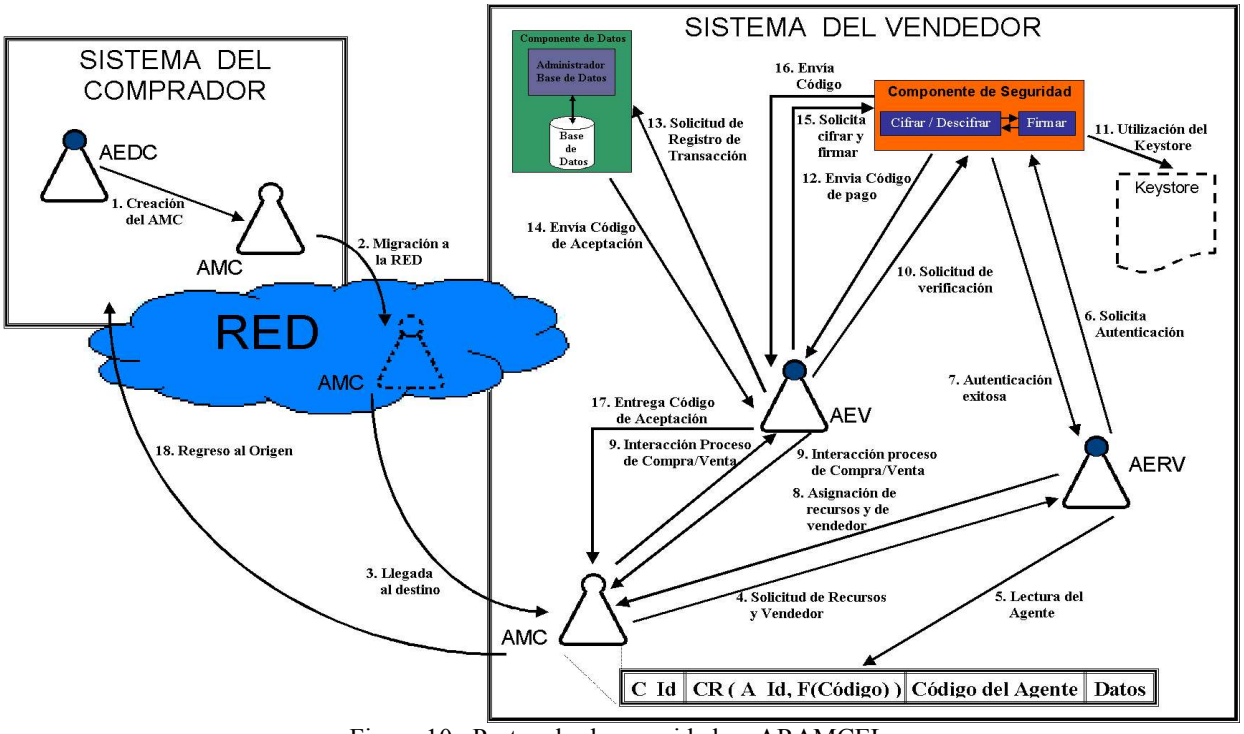


Figura 10. Protocolo de seguridad en ARAMCEL.

El AMC entrega el código de pago, cifrado y firmado al Agente Estacionario Vendedor (AEV). El AEV envía el código de pago al Componente de Seguridad y le solicita la verificación de la firma. Ese componente utiliza la información almacenada en el Keystore para descifrar el código de pago y verificar la firma; después de esto, devuelve el código de pago descifrado al AEV. Este último hace la solicitud de registro de la transacción al Componente de Datos, el cual genera y almacena el código de aceptación y envía al AEV. A su vez el AEV solicita al Componente de Seguridad el cifrado y firmado del código, cuando esto se ha llevado a cabo, el código es entregado al AMC, quién regresa a su origen.

La validación de este trabajo de investigación se realizó por medio del prototipo ADAM: Aplicación de los agentes móviles al comercio electrónico [3], al cual se le hicieron pruebas de verificación, validación y rendimiento.

Conclusiones

Los agentes móviles debido a las ventajas que poseen presentan fortalezas para trabajar de manera eficiente en sistemas distribuidos, pero también tienen inconvenientes de seguridad; los cuales son una oportunidad al generar un campo de acción en investigación para generar soluciones.

Con ARAMCEL se presenta una propuesta para crear aplicaciones que con base en las especificaciones dadas, aprovechen las ventajas de los agentes en el Comercio Electrónico; así mismo, ARAMCEL pretende dar un aporte a las necesidades de un mundo en proceso de globalización y que a su vez provea un entorno seguro de ejecución para ese tipo de sistemas.

A lo largo de este artículo, se han discutido las amenazas de seguridad existentes en un sistema para Comercio Electrónico basado en agentes móviles, los problemas del agente hostil y servidor hostil. Con base en esos ataques, en los aspectos de seguridad autenticación, autorización y no-repudio y en PKI se presentó el modelo de seguridad de ARAMCEL que ofrece una solución a esos problemas.

Aunque se ha ofrecido una solución a algunos de los problemas de seguridad, existen otros que son objeto de investigación en el mundo, además de aquellos que se presenten a medida que los sistemas basados en agentes móviles para Comercio Electrónico se hagan más frecuentes. Sin embargo, considerando las características del modelo de seguridad presentado es posible extenderlo y enriquecerlo para contrarrestar nuevos tipos de ataques que surjan en el entorno.

Agradecimientos

El segundo autor desea expresar su agradecimiento por la colaboración y apoyo recibido de la Universidad del Valle - Sede Tuluá.

Referencias

[1] Dávila, J. El Comercio Electrónico todavía está por llegar. Asociación Española de Criptología y Seguridad de la Información. España. 2000.

[2] Desouza, K. Intelligent Agents for Competitive Intelligence: Survey of Applications. University of Illinois at Chicago. US. 2001

[3] Gómez, J.; Castillo, S. (Director) y León, L. (Codirector). ADAM: Aplicación de los Agentes Móviles al Comercio Electrónico. Tesis de Pregrado (Ingeniería de Sistemas). Universidad Industrial de Santander. Colombia. 2004.

[4] Hidekazu, T. Security Technology for Electronic Commerce. Electronic Commerce Promotion Council. Japan. 2000.

[5] Jansen, W. and Karygiannis, T. Mobile Agent Security. National Institute of Standards and Technology. Computer Security Division. US. 1999.

[6] Kabay, M. A Primer on Public Key Infrastructures. Program Master of Science in Information Assurance. Division of Business & Management. Norwich University. US. 2004

[7] Lange, D. and Oshima, M. Programming and Deploying Java Mobile Agents with Aglets. Addison Wesley. US. 1998.

[8] Man M.C. and Wei V.K. A Taxonomy for Attacks on Mobile Agent. IEEE EUROCON'2001 Trends in Communications, International Conference. 2001.

[9] Mascagni, M. Complexity and Analysis of Data Structures and Algorithms. Department of Computer Science. Florida State University. US. 2004.

[10] Oaks, S. Java Security, Second Edition. O'reilly & Associates, Inc. US. 2001.

[11] Smart, N. Public Key Infrastructure. Department of Computer Science. University of Bristol, UK. 2004

[12] Vigna G. Mobile Agents and Security, Lecture notes in Computer Science. Springer-Verlag, 1998.

[13] Zhao, J. and Blum, T. Next-Generation E-Commerce: XML + Mobile Agent + Trust. Fraunhofer CRCG, Providence, Rhode Island, USA. 2000.

Convergence Through a Weak Consistency Model: Timed Causal Consistency

Francisco J. Torres-Rojas
Centro de Investigación en Computación e Informática Avanzada (*CIenCIA*)
and Instituto Tecnológico de Costa Rica
torres@ic-itcr.ac.cr

and

Esteban Meneses
Centro de Investigaciones en Computación (*CIC*) - Instituto Tecnológico de Costa Rica
and PrediSoft
emeneses@ic-itcr.ac.cr

Costa Rica

Abstract

Given a distributed system with several shared objects and many processes concurrently updating and reading them, it is convenient that the system achieves convergence on the value of these objects. Such property can be guaranteed depending on the consistency model being employed. *Causal Consistency* is a weak consistency model that is easy and cheap to implement. However, due to the lack of real-time considerations, this model cannot offer convergence. A solution for overcoming that problem is to include time aspects within the framework of the model. This is the aim of *Timed Causal Consistency*.

Keywords: Convergence, Weak Consistency, Causal Consistency, Timed Consistency.

1 INTRODUCTION

A distributed application is build up from processes executed at different nodes from possibly distant locations. One important and not trivial problem arises when we want to maintain the consistency of the state shared by such processes. Many consistency protocols have been developed to attack this challenge. Each category is characterized by the type of executions that is permitted or banned by the protocol. However, in some applications it may be not necessary to require all the properties offered by strict consistency models. In its place, in could be better to implement less restrictive (and cheaper) protocols that satisfy the minimum requirements of the particular application.

Causal Consistency [2] is a weak protocol that only accepts executions where the causal order is respected. This model can be implemented at low cost. Besides this, although weak protocols are harder for programmers than strict ones, causal consistency offers a balance where a class of practical programs can be modeled and some useful properties guaranteed.

Convergence is an idea found in multiple areas of science. It is often related to some kind of stability. Although operations made in the past could have created certain disorganization in the arrangement of the analyzed system, there is comfort in knowing that our object of study achieves a stable behavior after a given instant. Consider, for instance, economical indexes after a financial crisis, or descriptive variables in the weather after a hurricane. It is nice to know that, after some time, prices will be under control, and that sunny afternoons may be enjoyed. These conditions will be "well behaved" until some other set of events strikes our system. At least for some range of time we are capable of understanding the state of our system.

In groupware systems (e.g., a collaborative editing system where many users are concurrently working over a document), it is fundamental to offer convergence. In this context, some authors [4, 13] define convergence by looking at the final result of a work session. Operations made by users could arrive at different times to the other sites, executing possibly in different orders. However, it is required that the final result be exactly the same for every user. It is also important offering convergence in mobile computing applications

[6], specially when disconnection periods are considered. In this case, after operations (possibly conflicting) are made over different replicas of the same object, it is required that all replicas converge to the same state after all the processes have been reconnected for sufficiently long.

Nevertheless, convergence is **not** a property inherent to causally consistent executions. *Timed Causal Consistency* is a protocol where time constraints are added to the causal consistency model. Ordering and timeliness are two facets of consistency protocols ([3, 15]). The ordering aspect defines the possible orders in which operations can be executed and perceived by the participant sites, while the timeliness defines how soon the effects of a operation in some process are known by the other processes.

In Section 2, the principal concepts of consistency protocols are revisited. The timed strategy is presented in Section 3, while the convergence model is developed in Section 4. Analysis about convergence and timed causal consistency is provided in Section 5. Conclusions and future work were left for Section 6.

2 CONSISTENCY MODELS REVISITED

A distributed system consists of N user processes and a distributed data storage. Because of caching and replication, several, possibly different, copies of the same data objects might coexist at different sites of the system. Thus, a consistency model, understood as a contract between processes and the data storage, must be provided. There are multiple consistency models [1, 2, 3, 8, 10, 14, 15].

The *global history* $\mathcal{H}$ of this system is the partially ordered set of all operations occurring at all sites. $\mathcal{H}_i$ is the total ordered set or sequence of operations that are executed on site i. If **a** occurs before **b** in $\mathcal{H}_i$ we say that **a** precedes **b** in *program order*, and denote this as $\mathbf{a} <_{\texttt{PROG}} \mathbf{b}$. In order to simplify, we assume that all operations are either **read** or **write**, that each value written is unique, and that all the objects have an initial value of zero. These operations take a finite, non-zero time to execute, so there is a time elapsed from the instant when a **read** or **write** "starts" to the moment when such operation "finishes". Nevertheless, for the purposes of this paper, we associate an instant to each operation, called the *effective time* of the operation. We will say that **a** is *executed* at time t if the effective time of **a** is t. If **a** has an effective time previous to the effective time of **b** we denote this as $\mathbf{a} <_{\texttt{E-T}} \mathbf{b}$. Let $\mathcal{H}_{i+w}$ be the set of all the operations in $\mathcal{H}_i$ plus all the **write** operations in $\mathcal{H}$. The partially ordered *happens-before* relationship "$\rightarrow$" for message passing systems as defined in [9] can be modified to order the operations of $\mathcal{H}$. Let **a**,**b** and $\mathbf{c} \in \mathcal{H}$, we say that $\mathbf{a} \rightarrow \mathbf{b}$, i.e., **a** happens-before (or *causally precedes*) **b**, if one of the following holds:

1. **a** and **b** are executed on the same site and **a** is executed before **b**.
2. **b** reads an object value written by **a**.
3. $\mathbf{a} \rightarrow \mathbf{c}$ and $\mathbf{c} \rightarrow \mathbf{b}$.

If $\mathcal{D}$ is a set of operations, then a *serialization* of $\mathcal{D}$ is a linear sequence S containing exactly all the operations of $\mathcal{D}$ such that each **read** operation to a particular object returns the value written by the most recent (in the order of S) **write** operation to the same object. If $\prec$ is an arbitrary partially ordered relation over $\mathcal{D}$, we say that serialization S *respects* $\prec$ if $\forall$ **a**, $\mathbf{b} \in \mathcal{D}$ such that $\mathbf{a} \prec \mathbf{b}$ then **a** precedes **b** in S.

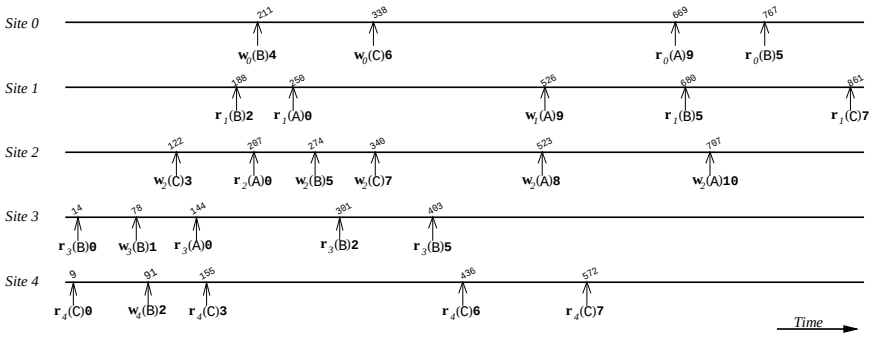
Intuitively, one would like that any **read** on a data item `X` returns a value corresponding to the results of the most recent **write** on `X`. In some systems this could mean that after making an update, all other processes may be notified about the change as soon as it is required. Assuming the existence of absolute global time, this behavior can be modeled with *linearizability* [8]:

Definition 1 *History* $\mathcal{H}$ *satisfies* Linearizability (**LIN**) *if there is a serialization* S *of* $\mathcal{H}$ *that respects the order* $<_{\texttt{E-T}}$ [8].

A weaker, but more efficient, model of consistency is *sequential consistency* as defined by Lamport in [10]:

Definition 2 *History* $\mathcal{H}$ *satisfies* Sequential Consistency (**SC**) *if there is a serialization* S *of* $\mathcal{H}$ *that respects the order* $<_{\texttt{PROG}}$ *for every site in the system* [10].

SC does not guarantee that a **read** operation returns the most recent value with respect to real-time, but just that the result of any execution is the same as if the operations of all sites were executed in some sequential order, and the operations of each individual site appear in this sequence in the order specified by its program. For instance, History $\mathcal{H}$ presented in 1.**a)** is sequentially consistent, because 1.**b)** shows the required serialization S. Although **SC** can be implemented in a more efficient way than **LIN** and it is a programmer-friendly model, it has been shown that **SC** has performance problems [1, 14].

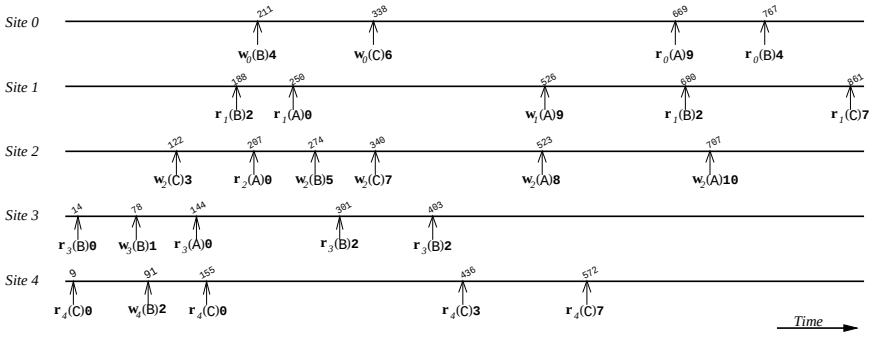


a) Sequentially Consistent Execution

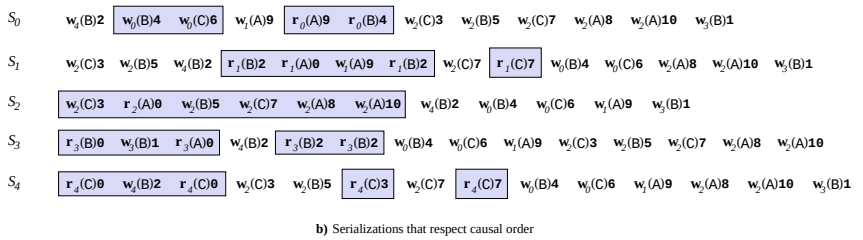
$r_4(C)0\ r_3(B)0\ w_0(B)4\ w_2(C)3\ r_2(A)0\ w_3(B)1\ r_3(A)0\ w_4(B)2\ r_4(C)3\ r_3(B)2\ r_1(B)2\ r_1(A)0\ w_0(C)6\ w_1(A)9\ r_0(A)9\ w_2(B)5\ r_1(B)5\ r_0(B)5$
$r_3(B)5\ r_4(C)6\ w_2(C)7\ r_4(C)7\ r_1(C)7\ w_2(A)8\ w_2(A)10$

b) Serialization that respects program order

Figure 1: Distributed history compliant with sequential consistency



a) Causally Consistent Execution



b) Serializations that respect causal order

Figure 2: A causally consistent distributed history

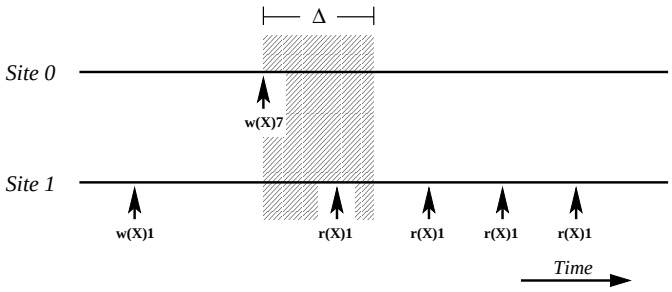


Figure 3: A non-timed sequentially consistent execution

An even weaker model of consistency is *causal consistency* [2]. First, let $\mathcal{H}_{i+w}$ be the set of all the operations in $\mathcal{H}_i$ plus all the **write** operations in $\mathcal{H}$. Second, we modify the *happens-before* relationship "$\rightarrow$" for message passing systems as defined in [9] to order the operations of $\mathcal{H}$. Let **a**,**b** and **c** $\in \mathcal{H}$, we say that **a** $\rightarrow$ **b**, i.e., **a** happens-before (or *causally precedes*) **b**, if one of the following holds:

1. **a** and **b** are executed on the same site and **a** is executed before **b**.
2. **b** reads an object value written by **a**.
3. **a** $\rightarrow$ **c** and **c** $\rightarrow$ **b**.

Two distinct operations **a** and **b** are *concurrent* if none of these conditions hold between them.

Definition 3 *History* $\mathcal{H}$ *satisfies* Causal Consistency (**CC**) *if for each site* i *there is a serialization* S_i *of the set* $\mathcal{H}_{i+w}$ *that respects causal order "*$\rightarrow$*"* [2].

Thus, if **a**,**b** and **c** $\in \mathcal{H}$ are such that **a** writes value **v** in object `X`, **c** reads the same value **v** from object `X`, and **b** writes value **v'** into object `X`, it is never the case that **a** $\rightarrow$ **b** $\rightarrow$ **c**. **CC** requires that all causally related operations be seen in the same order by all sites, while different sites could perceive concurrent operations in different orders. For example, if figure 2 we have a distributed history in 2.**a** which is causally consistent, due to serializations S_i presented in 2.**b**. Note that in 2.**b** there is a serialization S_i for site i and the local events in S_i are distinguished by a surrounding rectangle.

CC is a model of consistency weaker than **SC**, but it can be implemented efficiently [2, 14]. Such implementation requires keeping track of which processes have seen which **write** events. In fact, there is dependency graph for determining which operation is dependent on which other operations. So, this data structure must be built and maintained. For fulfilling this need, vector clocks [11] can be used.

3 TIMED CONSISTENCY MODEL

In neither **SC** nor **CC** real-time is explicitly captured, i.e., in the serializations of $\mathcal{H}$ or $\mathcal{H}_{i+w}$ operations may appear out of order in relation to their effective times. For instance, in Figure 1 the serialization in part **b)** shows event $\mathbf{r}_1(\mathtt{B})\mathbf{5}$ occurring before $\mathbf{w}_2(\mathtt{A})\mathbf{8}$, but the latter event occurred at time 523, while the former occurred at time 680. In **CC**, each site can see concurrent **write** operations in different orders. On the other hand, **LIN** requires that the operations be observed in their real-time ordering. Ordering and time are two different aspects of consistency. One avoids conflicts between operations, the other addresses how quickly the effects of an operation are perceived by the rest of the system.

Timed consistency (**TC**) as proposed in [15] requires that if the effective time of a **write** is t, the value written by this operation must be visible to all sites in the distributed system by time $t + \Delta$, where Δ is a parameter of the execution. It can be seen that when $\Delta = 0$, then **TC** becomes **LIN**. So, **TC** can be considered as a generalization or weakening of **LIN**.

The execution showed in Figure 3 satisfies **SC** and **CC**. Up to the second operation of Site *1*, the execution satisfies **TC** for the value of Δ presented in this figure, but, by that same instant, **LIN** is no longer satisfied. After this point, the execution is not even timed because there are **read** operations in Site *1* that start more than Δ units of real-time after Site *0* writes the value **7** into object `X` and these **read** operations do not return this value.

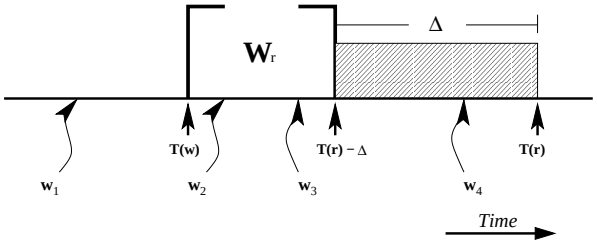


Figure 4: Operation **r** does not read on time

3.1 Reading on Time

In *timed* models, the set of values that a **read** may return is restricted by the amount of time that has elapsed since the preceding **writes**. A **read** occurs *on time* if it does not return stale values when there are more recent values that have been available for more than Δ units of time. This definition depends on the properties of the underlying clock used to assign timestamps to the operations in the execution. Let $T(\mathbf{a})$ be the real-time instant corresponding to the effective time of operation **a**.

Definition 4 *Let $\mathcal{D} \subseteq \mathcal{H}$ be a set of operations and S a serialization of $\mathcal{D}$. Let* **w**, **r** $\in \mathcal{D}$ *be such that* **w** *writes a value into object* `X` *that is later read by* **r**, *i.e.,* **w** *is the closest* **write** *operation into object* `X` *that appears to the left of* **r** *in serialization S. We define the set $\mathcal{W}_{\mathbf{r}}$, associated with* **r**, *as: $\mathcal{W}_{\mathbf{r}}$=*{**w'** $\in \mathcal{D}$ | (**w'** *writes a value into object* `X`*)* $\wedge$ *(T(**w**) < T(**w'**) < T(**r**) - Δ)}. *We say that operation* **r** ***occurs*** *or* ***reads on time*** *in serialization S, if $\mathcal{W}_{\mathbf{r}} = \varnothing$. S is* **timed** *if every* **read** *operation in S occurs on time.*

Figure 4 illustrates Definition 4, presenting a possible arrangement of **read** and **write** operations over the same object. Operation **r** reads a value previously written by operation **w**. Since operation $\mathbf{w}_1$ was executed before **w**, it has no effect on whether **r** is reading on time or not. Similarly, although $\mathbf{w}_4$ is more recent than **w**, the interval Δ has not elapsed yet when **r** is executed, and, thus, it is acceptable that **r** does not observe the value written by $\mathbf{w}_4$. On the other hand, operations $\mathbf{w}_2$ and $\mathbf{w}_3$ occur after **w**, and the values written by then have been available in the system for more than Δ units of time when **r** is executed. Thus, $\mathbf{w}_2$ and $\mathbf{w}_3$ are in $\mathcal{W}_{\boldsymbol{r}}$, and, therefore, operation **r** does not occur on time. The area between $T(\mathbf{w})$ and $T(\mathbf{r})$-Δ represents the interval of time associated with the set $\mathcal{W}_{\boldsymbol{r}}$, which according to definition 4 must be empty if **r** reads on time (i.e. no write operation to the same object read by **r** can occur in this interval).

Definition 5 *Let* **a**, **b** $\in \mathcal{D} \subseteq \mathcal{H}$ *with effective times t_1 and t_2, respectively, be two operations over the same object* `X`. *We say that* **a** $<_\Delta$ **b** *if:*

1. *Both* **a** *and* **b** *are* **write** *operations and $t_1 < t_2$, or*
2. **a** *is a* **write** *operation,* **b** *is a* **read** *operation and $t_1 < (t_2 - \Delta)$.*

Definition 6 *History $\mathcal{H}$ satisfies* Timed Consistency (**TC**) *if there is a serialization S of $\mathcal{H}$ that respects the partial order $<_\Delta$* [15].

3.2 Timed Sequential Consistency and Timed Causal Consistency

Now, we combine the requirements of well-known consistency models such as **SC** and **CC** with the requirement of reading on time.

Definition 7 *History $\mathcal{H}$ satisfies* Timed Sequential Consistency (**TSC**) *if there is a serialization S of $\mathcal{H}$ that simultaneously respects the partial order $<_{PROG}$ and the partial order $<_\Delta$* [15].

Definition 8 *History $\mathcal{H}$ satisfies* Timed Causal Consistency (**TCC**) *if for each site* i *there is a timed serialization S_{i} of $\mathcal{H}_{\mathrm{i}+\boldsymbol{w}}$ that simultaneously respects causal order $\rightarrow$ and the partial order $<_\Delta$* [15].

Figure 5 presents a hierarchy of different consistency models. Every sequentially consistent execution is also linearizable, but the opposite is not necessarily true. Similarly, every causally consistent execution is sequentially consistent, while the contrary is not always true. The proofs for these results and some implementation details can be found in [15]. The idea behind these definitions is to show how it is possible to offer flexibility for the consistency protocol in our model; without losing the real-time considerations, consistency aspects can be relaxed, passing from **SC** to **CC**.

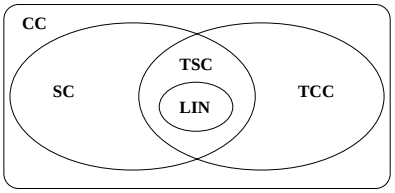


Figure 5: Hierarchy of consistency models

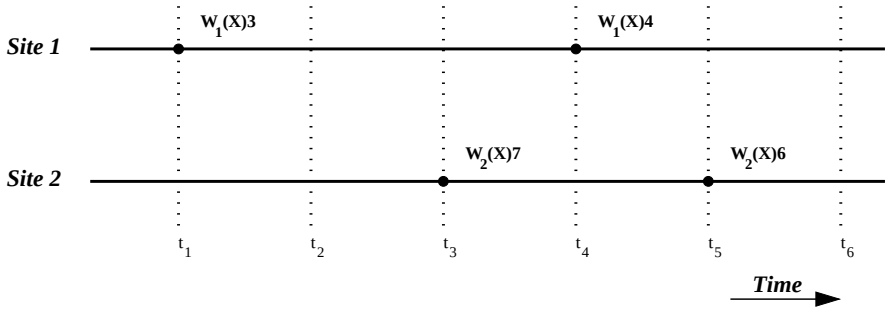


Figure 6: Two sites in a convergent execution

4 CONVERGENCE MODEL

In this section, we present our approach for analyzing convergence in consistency protocols for distributed systems.

Consider the distributed history shown in Figure 6. After `X` is updated by one of the sites, the new value is communicated, with some delay, to the other site. Site *1* updates `X` at time t_1 giving it the value `3` (which was `0` initially). By that time, Site *2* has no knowledge of this change, so it surely believes that `X` still has `0` value. It is not until time t_2 that Site *2* discovers that `X` has been updated. However, at time t_3 Site *2* makes a new change to `X`, giving it the value of `7`. Let's say that news of this change arrive too late to Site *1*, and by time t_4, Site *1* has updated again `X` to value `4`. Similarly, Site *2* does not perceive this last change and updates `X` at time t_5 to value `6`. At time t_6 Site *1* realizes that `X` has a new value and from here on, both sites agree on the value of `X`. Thus, finally, convergence has been reached.

Figure 7 plots the values that `X` takes at every instant, as perceived by each site, and it shows that there is convergence after time t_6. Nevertheless, it could be claimed that the time intervals $[0,t_1]$,$[t_2,t_3]$ and $[t_6,+\infty]$ form a set of ranges where convergence was achieved, we refer to these time ranges as *convergence frames* (see Section 4.3). Our formal definition for convergence will capture these two kinds of stability ranges: the

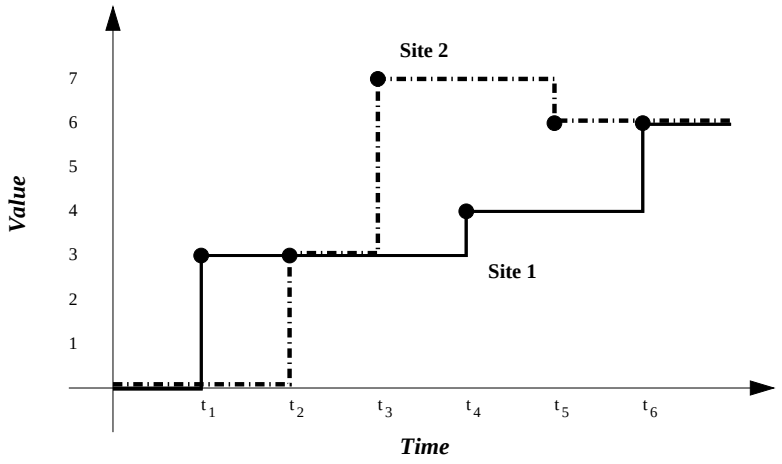


Figure 7: Convergence of two sites

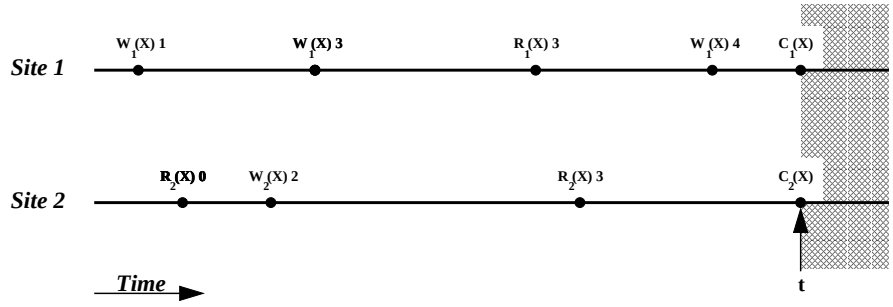


Figure 8: Convergent cut

one obtained after the *last* update to some distributed object, and the time frames where two or more sites agree on the same value for a particular object.

4.1 Trivial Convergence

It is possible that a distributed system achieves convergence over some object `X` if all sites have previously agreed in setting a particular value for such object. This is the less interesting case for convergence, since a situation like this hardly represents the general case, and even though the system is reaching convergence in the value of `X`, this does not imply correctness in the execution.

Definition 9 *An execution in a distributed system is* **trivially convergent over object** `X` *if all sites have agreed to assign a particular value for* `X` *after time t. If the execution is trivially convergent over every possible object, we say that the execution is* **trivially convergent**.

4.2 Absolute Convergence

Let's analyze convergence after the "last" **write** operation to a particular object. Figure 8 shows a simple distributed computation, with 2 sites and one shared object `X`. There are a series of **writes** and **reads** executed by both sites, and, at several times during execution, sites see different values for object `X`. But, at some time t after event $\mathbf{w}_1(\mathtt{X})4$, which happens to be the last actualization to `X` in the whole execution, all **reads** to object `X` executed by any site *should* return the value `4`. Thus, after time t, this system has converged regarding the value of object `X`. The intuition behind *absolute convergence* is that, at the end of the day, after the **writes** stop, every site involved in a distributed computation will agree on the same values for the same objects.

Following the lines of *consistent cuts* [11], we define a *convergent cut* this way:

Definition 10 *We say that a* **convergent cut over object** `X` *is a set of phantom events* $\mathcal{C}=\{C_1,C_2,...,C_N\}$, *where every* C_i *is inserted in local history* $\mathcal{H}_i$, *all at the same time t. All the* C_i *are* **read** *operations over* `X` *that would return exactly the value written by the latest* **write** *into object* `X` *that occurred before t.*

Definition 11 *An execution in a distributed system is* **absolutely convergent over object** `X` *if at any arbitrary time after t, which itself occurs after the last* **write** *to object* `X`, *a* convergent cut over object `X` *can be inserted. If the execution is absolutely convergent over every possible distributed object, we say that the execution is* **absolutely convergent**.

If an execution is absolutely convergent over object `X` at time t, i.e., we were able to insert a convergent cut $\mathcal{C}$ at time t, it must be true that the same cut $\mathcal{C}$ can be inserted, with identical results, at any time $u > t$. Now, if there are M shared objects $\mathtt{X}_j$, $1 \leq j \leq M$, and the execution is absolutely convergent for all the M objects, then for every object we associate a minimum time t_j where its corresponding convergence cut can be inserted. Therefore, the distributed system is absolutely convergent at any time after $t = max(t_1, t_2, ..., t_M)$.

4.3 δ-Convergence

It is typically desirable that the lapse before an update is communicated to everybody else in a distributed system be as short as possible. However, in a very active system with frequent **writes** to the same shared objects, it is normal that the values of these objects diverge during execution. Even under these circumstances, we could expect that after the "last" **write**, as mentioned in the previous section, the system reaches *absolute*

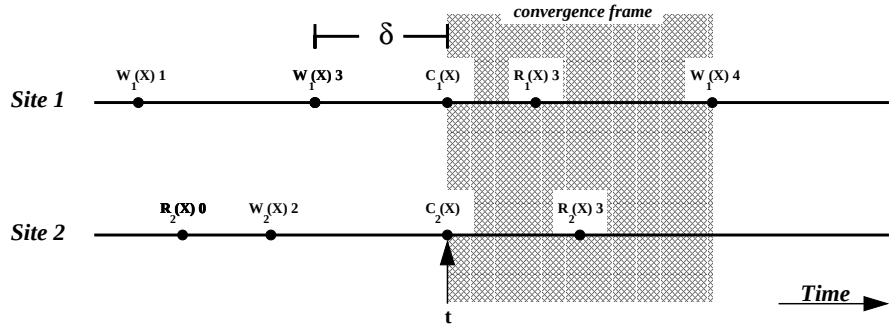


Figure 9: Convergence frame

convergence. Besides, if the system, after considering factors such as overhead, communication delays, and consistency protocols, can guarantee that an update is known to the complete system (either by updating or by invalidations) in at most δ units of time, the execution might manifest intervals where the system is evidently convergent in relation to some objects.

Now, we claim that if the lapses between multiple consecutive **writes** to the same object are shorter than the parameter δ, there was not enough time for propagating the values set by all the **writes**, but the system can still be classified as convergent. Conversely, if two consecutive **writes** to the same object `X` occur more than δ units apart and we are **not** able to insert a convergent cut for this object at least δ units of times after the first **write**, the system is not convergent. This is the intuition of what we called δ-convergence:

Definition 12 *An execution satisfies* **δ-convergence** *if it can be guaranteed that, at any time when the lapse between two consecutive* **writes** *to the same object* `X` *is greater than δ units of time, a convergent cut over* `X` *can be inserted δ units of time after the first* ***write****.*

Thus, in a δ-convergent execution, if `X` is updated at time t and the next update to this object, anywhere in the system, occurs at time u, with $t + \delta < u$, there is an interval $[t + \delta, u]$ where all sites in the system would perceive, if they read it, the very same value for object `X`. We call this interval a *convergence frame* for object `X`. On the other hand, if $t + \delta > u$, we might not define such a convergence frame, but still claim that the system is δ-convergent. In other words, the system is allowed to be "unstable" for at most δ units of time after a **write**, without being considered non-convergent.

Figure 9 shows some of the previous concepts. If δ units of time after operation $\mathbf{w}_1(\mathtt{X})\mathtt{3}$ occurred, we are able to insert a convergent cut associated to object `X` (which means that if every site in the system would read `X` all they would find the same value), this establishes a convergence frame for object `X`. Of course, another update to object `X` can be made thereafter, but until that new update the system has converged on the value of `X`. Extending this concept to several objects is straightforward.

5 CAUSAL CONSISTENCY AND CONVERGENCE

As it was mentioned in definition 3, **CC** does not require that each **read** event over object X returns the latest value written. It merely needs to construct serialization S_i, for each site i, that respects causal order "$\rightarrow$". Thus, **CC** does not offer convergence *per se*. Due to the lack of real-time restrictions in its definition, sites are not obliged to update or invalidate their local objects unless that it is required for building the serializations S_i.

Figure 10 presents a simple example of a distributed execution that satisfies **CC**, but whose shared objects never converge. Notice that this history is not compliant with **SC** either. Site *2* writes value `1` into object `X`, and, some time later, Site *1* writes value `2` into object `X`. The following **read** on Site *1* returns `1`, while the next **read** operation on Site *2* returns value `2`. Finally, Site *1* writes value value `3` into object `X`.

Considering only these five events, we prove that the history, so far, satisfies **CC** by taking the following serializations: $S_1 = \mathbf{w}_1(\mathtt{X})\mathtt{2}, \mathbf{w}_2(\mathtt{X})\mathtt{1}, \mathbf{r}_1(\mathtt{X})\mathtt{1}, \mathbf{w}_1(\mathtt{X})\mathtt{3}$ and $S_2 = \mathbf{w}_2(\mathtt{X})\mathtt{1}, \mathbf{w}_1(\mathtt{X})\mathtt{2}, \mathbf{r}_2(\mathtt{X})\mathtt{2}, \mathbf{w}_1(\mathtt{X})\mathtt{3}$. These serializations do not respect real-time, but fulfill the requirements for **CC** [2]. Now, nothing forces Site *1* nor Site *2* to agree, at any point in the future, on the value of `X`. Consider operation sets $\mathcal{Q}_1$ and $\mathcal{Q}_2$, both containing just **reads** on object `X`, one executing on Site *1* and the other executing on Site *2*, respectively. The value retrieved by operations in $\mathcal{Q}_1$ is `3`, while the old value retrieved by operations in $\mathcal{Q}_2$ is `2`. It can be proved by induction over $|\mathcal{Q}_1|$ and $|\mathcal{Q}_2|$ that this distributed history is causally consistent. A possible serialization set would be: $S_1 = \mathbf{w}_1(\mathtt{X})\mathtt{2}, \mathbf{w}_2(\mathtt{X})\mathtt{1}, \mathbf{r}_1(\mathtt{X})\mathtt{1}, \mathbf{w}_1(\mathtt{X})\mathtt{3}, \{\mathcal{Q}_1\}$ and $S_2 = \mathbf{w}_2(\mathtt{X})\mathtt{1}, \mathbf{w}_1(\mathtt{X})\mathtt{2}, \mathbf{r}_2(\mathtt{X})\mathtt{2}, \{\mathcal{Q}_2\}, \mathbf{w}_1(\mathtt{X})\mathtt{3}$.

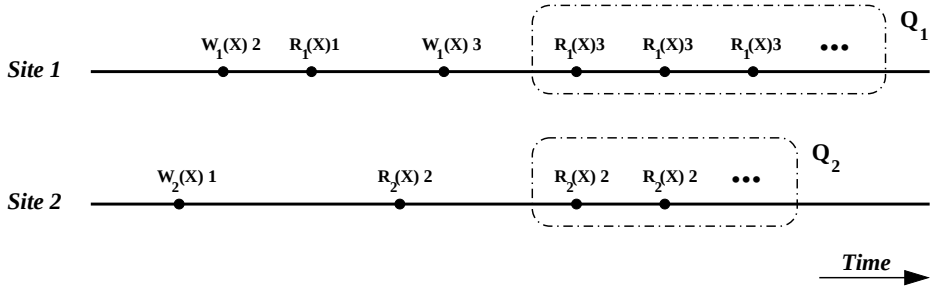


Figure 10: A causally consistent history that doesn't converge

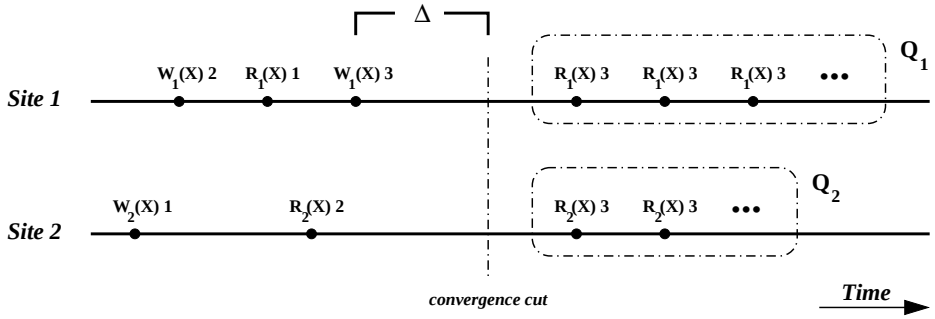


Figure 11: A timed causally consistent history

Thus, Sites *1* and *2* can execute an arbitrarily large number of **reads** of object `X`, satisfying **CC**, and never converging to the same value. In this example, *absolute convergence* is not guaranteed by **CC**, and, if we choose $\mathbf{w}_1(\mathtt{X})\mathtt{2}$ and $\mathbf{w}_2(\mathtt{X})\mathtt{1}$ as occurring more than δ units of time apart, neither δ-convergence is satisfied. It is easy to build an example more complex than Figure 10, involving multiple sites, shared objects and values written, where **CC** is respected, and where absolute convergence and δ-convergence are never met.

On the other hand, we claim that **TCC** guarantees convergence. Remember that **TCC** requires that if the effective time of a **write** is t, the value written by this operation must be visible to all sites in the distributed system by time $t + \Delta$, where Δ is a parameter of the execution.

Figure 11 shows the same execution of figure 10 but includes the requirements of **TCC**. Then, it can be observed that Δ time units after the event $\mathbf{w}_1(\mathtt{X})\mathtt{3}$ Site *2* must be aware of the changes done over object X. Then, a convergent cut can be placed at this moment and, after that, the sets $\mathcal{Q}_1$ and $\mathcal{Q}_2$ will report the same read value `3`. We can generalize this concept into the next theorem.

Theorem 1 ***TCC*** *satisfies* absolute convergence *and* δ-convergence.

Proof At most Δ units of time after the last **write** for every shared object, **TCC** guarantees that the updated value is known to every site in the distributed system, therefore, we can insert a convergent cut over each shared object Δ units of time after the corresponding last **write** operation, which according to Definition 11 proves that the execution satisfies *absolute convergence*. Now, it should be easy to see that δ-convergence is guaranteed for the value $\delta = \Delta$.

Both, absolute convergence and δ-convergence, are desirable properties among distributed systems applications, such as collaborative software and mobile computing. Recent research in these areas ([7],[12]) show that **CC** is a good alternative for maintaining consistency. Then, **TCC** which is a strengthening of **CC**, is a good candidate to be applied in those contexts.

6 CONCLUSIONS AND FUTURE WORK

The *causal consistency* model gives a weakening of *sequential consistency*, making a difference between events that are potentially causally related and those that are not. Its implementation is easier and cheaper than more strict models (like **LIN** or **SC**). Although **CC** does not guarantee convergence property for the values

of the shared objects in the system, it is possible to enrich this model with time considerations. Besides this, a timed consistency model provides a mix of two facets of consistency: order and time. **TCC** can be conceived as a promising candidate to provide convergence at low implementation costs.

Many distributed system applications require convergence as one of their most important properties. In those areas, **TCC** can be introduced to provide such objective, assuring at the same time that causally related updates will be respected by all processes.

In the short term, we are building a distributed shared memory system where diverse consistency protocols, timed and not timed, are implemented, tested and compared.

References

[1] Adve, S. and Gharachorloo, K. *Shared Memory Consistency Models: A Tutorial*. Western Research Laboratory, Research Report 95/7, 1995.

[2] Ahamad, M. et al. *Causal memory: definitions, implementation and programming*. Distributed Computing. September, 1995.

[3] Ahamad, M. and Raynal, M. *Ordering and Timeliness: Two Facets of Consistency?*, Future Directions in Distributed Computing, 2003.

[4] Ellis, C.A. and Gibbs, S.J. *Concurrency Control in Groupware Systems*. In ACM SIGMOD'89 proceedings,pages 399-407, 1989.

[5] Ellis, C.A., Gibbs, S.J. and Rein, G.L. *Groupware: some issues and experiences*. Communications of the ACM, Vol 34(1), January, 1991.

[6] Guerraoui, R. and Hari, C. *On the Consistency Problem in Mobile Distributed Computing*. ACM POMC, 2002.

[7] Galli, R. and Luo Y. *Mu3D: A Causal Consistency Protocol for a Collaborative VRML Editor*. VRML, Monterrey, California, USA, 2000.

[8] Herlihy, M. and Wing, J. *Linearizability: A Correctness Condition for Concurrent Objects*. ACM Transactions on Programming Languages and Systems. Vol 12(3), July 1990.

[9] Lamport, L. *Time, Clocks and the Ordering of Events is a Distributed System*. Communications of the ACM, vol 21, July, 1978.

[10] Lamport, L. *How to make a Multiprocessor Computer that correctly executes Multiprocess Programs*. IEEE Transactions on Computer Systems, C-28(9), 1979.

[11] Mattern, F. *Virtual Time and Global States of Distributed Systems*. Proceedings of the International Workshop on Parallel and Distributed Algorithms, 215-226, 1989.

[12] Ram, D. Janaki et al. *Causal Consistency in Mobile Environment*. URL: http://lotus.iitm.ac.in.

[13] Sun, C. et al. *Achieving convergence, causality-preservation, and intention-preservation in real-time cooperative editing systems*. ACM Transactions in Computer-Human Interaction, 5(1):63-108, 1998.

[14] Torres-Rojas, F. J., Ahamad, M. and Raynal, M. *Lifetime Based Consistency Protocols for Distributed Objects*. Proc. 12th International Symposium on Distributed Computing, DISC'98, Andros, Greece, September 1998.

[15] Torres-Rojas, F. J., Ahamad, M. and Raynal, M. *Timed Consistency for Shared Distributed Objects*. Annual ACM Symposium on Principles of Distributed Computing PODC'99, Atlanta, Georgia, 1999.

Estudo do Teste de Mutação para a Linguagem Standard ML

Thaise Yano
Adenilso da Silva Simão
José Carlos Maldonado
Universidade de São Paulo, Instituto de Ciências Matemáticas e de Computação
São Carlos, Brasil, Caixa Postal 668 - CEP 13560-970
{tyano,adenilso,jcmaldon}@icmc.usp.br

Abstract

Functional programming languages, such as SML (Standard Meta Language), Haskell and Lisp, focus on rules and matching of patterns, in contrast to procedural languages in which programs are written as a sequence of instructions. Programs in functional languages may have errors due to the misunderstanding of their properties. Therefore, in this work, we establish mechanisms to investigate the aplicability of Mutation Testing for testing functional programs, written in SML. Mutation Testing is a test criterion that allows to evaluate the quality of a test set and to guide the generation of test sets. The existence of a tool to support this criterion is essential due to the large amount of information related to its application. The web tool PROTEUM/SML, developed with the aim of applying the Mutation Testing to SML, implements the mutation operators defined in this work.

Keywords: Mutation Testing, Functional Programming Language, Standard ML.

Resumo

Linguagens de programação funcionais, tais como SML (Standard Meta Language), Haskell e Lisp, enfatizam regras e casamento de padrões, ao contrário das linguagens procedimentais em que os programas são escritos como uma seqüência de instruções. Os programas em linguagens funcionais podem conter erros pela falta de entendimento de suas propriedades. Assim, neste trabalho, estabelecem-se subsídios para a investigação da aplicabilidade do Teste de Mutação para o teste de programas funcionais, escritos em SML. O Teste de Mutação é um critério de teste que fornece uma maneira de auxiliar na geração e na avaliação de um conjunto de casos de teste. Devido ao grande volume de informações que estão envolvidas na aplicação do Teste de Mutação, é essencial a existência de ferramentas de apoio para o uso desse critério. A fim de viabilizar a aplicação do Teste de Mutação para SML foi desenvolvida a ferramenta *web* PROTEUM/SML, que implementa os operadores de mutação definidos neste trabalho.

Palavras chaves: Teste de Mutação, Linguagem de Programação Funcional, Standard ML.

1 Introdução

Embora durante todo o processo de desenvolvimento de software sejam utilizadas técnicas, métodos e ferramentas a fim de evitar que erros sejam introduzidos no produto, a atividade de teste é de fundamental importância para a identificação e posterior eliminação dos erros que persistem, nas diversas fases do desenvolvimento. Além disso, a atividade de teste de software visa a fornecer evidências de confiabilidade e qualidade de um produto de software, em complemento a outras atividades, como por exemplo, o uso de revisões e de técnicas formais rigorosas de especificação e de verificação.

O Teste de Mutação, um critério de teste baseado em erros, procura revelar erros típicos cometidos no desenvolvimento de um produto. Um aspecto importante desse critério é fornecer uma maneira de auxiliar na geração e avaliação de um conjunto de casos de teste. Devido ao grande volume de informações que estão envolvidas na aplicação do Teste de Mutação, em que, geralmente, um grande número de produtos deve ser gerado, simulado e comparado, é essencial a existência de ferramentas de apoio para o uso desse critério. O desenvolvimento de ferramentas para o suporte à atividade de teste é de grande importância, uma vez que sua condução manual é propensa a erros, improdutiva e limitada a produtos muito simples.

Linguagens funcionais realizam computações por meio de definições e aplicações de funções [29]. Estas linguagens são livres de *side-effects*, não possuindo, portanto, comandos de atribuições e uma expressão sempre será avaliada com o mesmo valor. Segundo Claessen et al. [6], apesar de linguagens funcionais

apresentarem um estilo de programação que favorece o desenvolvimento de programas com menores taxas de erros, programas funcionais podem conter erros decorrentes da falta de entendimento de suas propriedades. Nesse contexto, é importante a investigação de critérios de teste para apoiar o teste de programas funcionais. Entretanto, existem poucas iniciativas para o teste de programas funcionais bem como no desenvolvimento de ferramentas que apóiem essa atividade [4, 13, 20, 21]. Além disso, tais iniciativas não possuem uma medida de cobertura da atividade de teste.

A linha de pesquisa deste trabalho é a investigação da adequabilidade da aplicação de critérios de teste tradicionalmente utilizados em programas imperativos no teste de programas funcionais. Em particular, busca-se estabelecer subsídios para a investigação da aplicação do Teste de Mutação para o teste de programas em SML (*Standard Meta Language*) [19]. A fim de viabilizar tal aplicação, foi desenvolvida a ferramenta PROTEUM/SML [31], que implementa os operadores de mutação para SML definidos por [30]. Os operadores de mutação caracterizam o Teste de Mutação para a linguagem/especificação alvo, estabelecendo os requisitos de teste a serem satisfeitos.

Este artigo está organizado da seguinte forma. Na seção 2 são apresentados os conceitos referentes ao Teste de Mutação e linguagens funcionais, enfatizando-se a linguagem SML. Na seção 2.2.1 são relacionados alguns trabalhos direcionados à VV&T (Verificação, Validação e Teste) de programas funcionais. Na seção 3 são descritos os operadores de mutação definidos para SML. Na seção 4 são apresentados as funcionalidades, a arquitetura e os aspectos de implementação relevantes da PROTEUM/SML. Na seção 5 é apresentado um estudo de caso para ilustrar a aplicação do Teste de Mutação para um programa SML. Finalmente, na seção 6 são feitas as considerações finais e sobre trabalhos futuros.

2 Conceitos Básicos

2.1 Teste de Mutação

O Teste de Mutação é um critério de teste baseado em erros, originalmente proposto para o teste de unidade [9]. Contudo, diversos pesquisadores têm aplicado seus conceitos fundamentais em vários contextos, tais como no teste de integração [8], teste de programas orientados a objeto [2, 14–16], teste de especificação [11, 12, 25, 28], teste de protocolo [22] e teste de modelo de segurança de rede [24]. Esse critério fornece ao testador um modo sistemático tanto para guiar a geração de casos de teste quanto para avaliar a qualidade do caso de teste. O Teste de Mutação utiliza produtos criados a partir do produto a ser testado (chamados de *mutantes*), com pequenas alterações sintáticas, que são modeladas por *operadores de mutação*. Em geral, os operadores estão associados a um tipo ou classe de erros que se pretende revelar no produto em teste.

O Teste de Mutação consiste em encontrar casos de teste que façam os mutantes comportarem-se diferentemente do produto original P, a fim de distinguir os mutantes. Os mutantes que foram distinguidos são ditos 'mortos'. Os mutantes 'vivos' são aqueles que se comportaram como o produto original para todo o conjunto T de casos de teste. Isso pode ocorrer i) porque o mutante é equivalente ao produto original ou ii) porque o conjunto de casos de teste não foi adequado o suficiente para distinguir o mutante. Na primeira situação, os mutantes podem ser desconsiderados. Na segunda situação, o conjunto T de casos de teste deve ser melhorado.

A adequação de um conjunto de casos de teste em relação a um programa em teste é obtida através do *escore de mutação*, que fornece a porcentagem de mutantes não-equivalentes mortos pelo conjunto de casos de teste, sendo definido da seguinte maneira:

$$ms(P,T) = \frac{DM(P,T)}{M(P) - EM(P)}$$

sendo:
$DM(P,T)$: número de mutantes mortos por T;
$M(P)$: número total de mutantes gerados; e
$EM(P)$: número de mutantes gerados equivalentes a P.

O escore de mutação fornece ao testador um mecanismo de avaliar a qualidade da atividade de teste. Quando o escore de mutação alcança valor igual a 1.00, diz-se que o conjunto T de casos de teste é adequado em relação ao Teste de Mutação para testar o produto P, aumentando a confiança no produto em teste.

2.2 Linguagens Funcionais

Linguagens funcionais enfatizam a avaliação de expressões, ao invés da execução de comandos. As expressões são formadas usando-se funções, que são consideradas como valores tais como inteiro e *string*. Uma função pode retornar outra função, pode utilizar funções como parâmetros e, até mesmo, construir composição de

funções. Não há *side-effects* em programas funcionais, assim o valor da avaliação de uma expressão sempre será o mesmo [29].

SML (*Standard Meta Language*) caracteriza-se, principalmente, por ser uma linguagem de programação funcional. Essa linguagem foi proposta em 1983 e, posteriormente, definida formalmente em notações matemáticas por [18]. Uma revisão da linguagem, conhecida como SML'97, foi feita para torná-la mais simples, sendo descrita em [19].

Uma conseqüência da programação funcional é que a computação ocorre através das avaliações de expressões e não por atribuições a variáveis. Porém, SML difere das linguagens funcionais puras no sentido de permitir o uso de construtores tais como variáveis e operações de atribuição.

Um aspecto importante de SML é ser uma linguagem fortemente tipada, na qual todos os valores e variáveis possuem um tipo que pode ser determinado em tempo de compilação, sem a necessidade de executar o programa. Isso garante que apenas operações compatíveis sejam realizadas [23]. Esse processo de checagem de tipo permite que muitos erros sejam encontrados pelo compilador, não sendo necessária a execução do programa. Embora a maioria das linguagens fortemente tipadas requeira uma declaração de tipo de todas variáveis, em SML isso não é necessário e apenas se requer uma declaração de tipo quando é impossível de se deduzir o tipo [29].

Além disso, SML apóia polimorfismo que permite uma função ter argumentos de vários tipos. Outra característica de SML, visando ao desenvolvimento de programas de larga-escala, é de fornecer facilidades como de modularidade, encapsulamento de conceitos e reuso de software, através de *structures* (conjunto de tipos de dados, funções, exceções e demais elementos), *signatures* (tipo para *structure*) e *functors* (função que possui como argumento *structures* e outros elementos, e retorna um *structure*) [29].

SML fornece um mecanismo de tratamento de exceções, sendo que as exceções podem incluir valores arbitrários, inclusive funções. Em qualquer ponto do código, uma exceção pode ocorrer, abortando a operação atual e retornando o controle para o último tratador definido para tal exceção [23].

2.2.1 Programas Funcionais: VV&T

A investigação de critérios de teste para apoiar o teste de programas funcionais é de grande importância para a revelação de erros decorrentes da falta de entendimento de suas propriedades. Entretanto, existem poucas iniciativas para o teste de programas funcionais bem como no desenvolvimento de ferramentas que dêem apoio a essa atividade.

Uma ferramenta que pode-se citar é a *QuickCheck* [4, 5] que apóia o teste de programas Haskell, uma linguagem de programação funcional pura. Tal ferramenta testa propriedades dos programas, as quais são descritas como funções Haskell. O teste pode ser feito automaticamente por entradas aleatórias ou pelo testador. Porém, com essa abordagem de teste aleatório, *QuickCheck* torna-se limitado a pequenos programas e não possui uma medida de cobertura que contribui para avaliar a qualidade e a adequação da atividade de teste.

HUnit [13] é um *framework* de teste de unidade para Haskell, similar à ferramenta JUnit para Java. Permite que casos de teste sejam estruturados hierarquicamente, podendo ser executados automaticamente.

Auburn [20] é uma ferramenta para *benchmarking* de estruturas de dados funcionais. Produz uma árvore que classifica as melhores estruturas de dados de acordo como é utilizado. Possui a capacidade de extrair um perfil de como uma aplicação usa uma estrutura de dados.

CaslTest [21] é uma ferramenta que auxilia no teste de especificações CASL utilizando a linguagem SML. CASL é uma linguagem unificada para descrever especificações algébricas. O objetivo da ferramenta é extrair casos de teste das especificações CASL e projetar e gerar os oráculos de teste correspondentes e dados de teste em SML. Dessa maneira, verificam-se se os casos de teste derivados dos axiomas da especificação são satisfeitos pelos programas escritos em SML. Entretanto, CaslTest também não estabelece nenhuma medida de cobertura do teste realizado.

Devido à ausência de critérios de teste que possuam medida de cobertura da atividade de teste em programas funcionais, neste trabalho busca-se fornecer subsídios para a investigação da aplicação do Teste de Mutação em programas SML. O Teste de Mutação, através do escore de mutação, permite obter a adequação de um conjunto de casos de teste em relação a um programa em teste. Na próxima seção são descritos os operadores de mutação para SML definidos por [30], os quais são implementados na ferramenta Proteum/SML, que oferece apoio ao Teste de Mutação para programas SML.

3 Operadores de Mutação para SML

A definição dos operadores de mutação compreende um dos aspectos mais importantes para a aplicação do Teste de Mutação. Os operadores caracterizam o critério, estabelecendo os requisitos de teste a serem satisfeitos. Em geral, o desenvolvimento do conjunto de operadores está relacionado com uma classe de

erros típicos que podem ser cometidos no contexto de uma determinada linguagem/especificação. Assim, os operadores de mutação devem representar a "implementação" dessa classe de erros.

Além dos tipos de erros que se desejam revelar e da cobertura que se quer garantir, a escolha de um conjunto de operadores de mutação depende também da linguagem em que os programas estão escritos. Por exemplo, em [3] encontra-se a relação de 22 operadores de mutação utilizados por um sistema de mutação para programas em Fortran. Para a linguagem C, Agrawal [1] define um conjunto de 71 operadores de mutação, os quais foram implementados na ferramenta Proteum/C [7].

Nesta seção é apresentado o conjunto de 17 operadores de mutação definido para a aplicação do Teste de Mutação para a linguagem SML e implementado na PROTEUM/SML. Os operadores são classificados em função do elemento no qual é realizado a mutação. A classificação e a quantidade de operadores definidos para cada classe são dadas a seguir: expressões (3), operadores (5), constantes (4) e variáveis (5). O domínio de um operador de mutação é estabelecido em termos da entidade sintática que é afetada. Os nomes dos operadores são formados por três letras maiúsculas, cuja primeira letra indica a classe do operador: **E** (*expression*), **O** (*operator*), **C** (*constant*) e **V** (*variable*), e as demais letras indicam a descrição do operador. Por exemplo, o operador VDT pertence a classe de variáveis e verifica a cobertura do domínio das variáveis (*Domain Trap*). Na Tabela 1 é apresentado resumidamente esse conjunto de operadores de mutação.

Tabela 1: Operadores de Mutação definidos na PROTEUM/SML

Operador	*Descrição*
EDL	*Expression **D**e**L**etion*
	Exclui expressões mas mantém o ponto-e-vírgula no final da expressão
EME	***M**atch **E**xpression Mutation*
	Adiciona a função `trap_on_match()` em cada *pattern* de *matches*
ETI	***T**rap on **I**f Condition*
	Substitui a condição c da expressão `if-then-else` pelas funções `trap_on_true(c)` e `trap_on_false(c)`
OAR	***A**rithmetic Operator **R**eplacement*
	Substitui cada operador aritmético (`+, -, *, /,div,mod`) por outro operador aritmético
ORR	***R**elational Operator **R**eplacement*
	Substitui cada operador relacional ($=, <, >, <=, >=, <>$) por outro operador relacional
OLR	***L**ogical Operator **R**eplacement*
	Substitui cada operador lógico (`andalso, orelse`) por outro operador lógico
OLN	***L**ogical **N**egation*
	Nega as expressões lógicas com `not`
OLC	***L**ogical **C**ontext Negation*
	Nega as condições da expressão `if-then-else` e `while-do` com `not`
CBR	***B**oolean **R**eplacement*
	Substitui a constante `true` por `false` e vice-versa
CCC	***C**onstant for **C**onstant Replacement*
	Troca cada constante (global/local) do programa por outras constantes (global/local)
CCS	***C**onstant for **S**calar Replacement*
	Troca cada variável (global/local) do programa por todas as constantes (global/local)
CRC	***R**equired **C**onstant Replacement*
	Troca cada variável por constantes requeridas
VAV	***A**nonymous **V**ariable*
	Substitui cada variável escalar (global/local) pela variável anônima
VDT	***D**omain **T**raps*
	Substitui cada variável pelas funções `trap_on_negative_integer(`x`)`, `trap_on_positive_integer(`x`)`, `trap_on_zero_integer(`x`)`, `trap_on_negative_real(`x`)`, `trap_on_positive_real(`x`)` `trap_on_zero_real(`x`)` `trap_on_true(`x`)`, `trap_on_false(`x`)`
VRF	***R**ecord **F**ield Mutation*
	Substitui o *label* do operador # por outros *labels* de registros
VSV	***S**calar **V**ariable Reference Replacement*
	Substitui cada variável escalar (global/local) por outra (global/local)
VTM	***T**widdle **M**utations*
	Substitui cada referência escalar x pelo predecessor imediato e o sucessor imediato do valor corrente do argumento, através das funções: `pred_integer(`x`)`, `succ_integer(`x`)`, `pred_real(`x`)` e `succ_real(`x`)`

Na Figura 1 são apresentados um programa em SML e um mutante gerado pela aplicação do operador de mutação ORR. Esse programa retorna verdadeiro se uma lista de caracteres é vazia ou é formada por letras ou números, caso contrário retorna falso. O operador ORR substitui cada operador relacional ($=, <, >, <=, >=, <>$) por outro operador relacional, a fim de verificar se a escolha do operador relacional está correta dentro de uma expressão.

Embora a linguagem SML apresente características de encapsulamento, polimorfismo e modularidade

```
fun valid_f nil = true
  | valid_f (y::ys) =
      if (((y >= #"A") andalso (y <= #"Z")) orelse
         ((y >= #"a") andalso (y <= #"z")) orelse
         ((y >= #"0") andalso (y <= #"9")))
      then valid_f(ys)
      else false;
```

Programa Original

```
fun valid_f nil = true
  | valid_f (y::ys) =
      if (((y > #"A") andalso (y <= #"Z")) orelse
         ((y >= #"a") andalso (y <= #"z")) orelse
         ((y >= #"0") andalso (y <= #"9")))
      then valid_f(ys)
      else false;
```

Mutante

Figura 1: Exemplo de Programa Mutante

através de *structures*, *signatures* e *functors*, não foram propostos operadores de mutação para tais aspectos. O conjunto de operadores de mutação definido por [30] é direcionado apenas para o teste de unidade, não sendo abrangente para o teste de integração.

A aplicação do conjunto de operadores de mutação em um programa P gera um conjunto $\phi(P)$ de mutantes. Um conjunto T de casos de teste é adequado a P com relação a $\phi(P)$, se para cada programa $M \in \phi(P)$, ou M é equivalente a P, e nesse caso M e P possuem o mesmo comportamento para todo domínio de entrada, ou M difere de P em no mínimo um ponto de teste. Para distinguir o comportamento do mutante M do programa original P, analisam-se suas saídas após a execução com o conjunto T de casos de teste. Considere uma expressão e em um programa P e e_m a mesma expressão mas contendo alguma mutação definida por um operador de mutação *op* que gera o mutante M. Em geral, um caso de teste t precisa satisfazer três condições para distinguir M e P [10]:

- Alcançabilidade: e_m precisa ser executado.
- Necessidade: O estado de M imediatamente após alguma execução de e_m precisa ser diferente do estado de P imediatamente após a execução de e.
- Suficiência: A diferença nos estados de P e M imediatamente após a execução de e_m e e precisa ser propagada até o final da execução de P e M tal que os estados finais alcançados entre eles quando executado com T sejam diferentes.

4 Proteum/SML

Proteum/SML é uma ferramenta *web* desenvolvida para dar apoio ao Teste de Mutação para programas em SML que implementa o conjunto de 17 operadores de mutação apresentado na Tabela 1.

4.1 Aspectos Operacionais

A Proteum/SML oferece um conjunto mínimo de operações que satisfaz os requisitos básicos de uma ferramenta de teste baseada em mutação:

- Tratamento de casos de teste: execução, inclusão/exclusão e habilitação/desabilitação;
- Tratamento de mutantes: geração, seleção, execução e análise; e
- Análise de adequação: escore de mutação e relatórios estatísticos.

A organização da Proteum/SML é feita por projetos e sessões, a qual permite minimizar o número de gerações de mutantes e execuções de casos de teste. Isso é de grande importância para a condução de estudos empíricos que utilizam um produto a ser testado com o mesmo conjunto de mutantes (ou um subconjunto desse) e/ou com casos de teste similares.

Cada projeto determina o código fonte do programa a ser testado e o conjunto de operadores que serão utilizados para a geração dos mutantes. Um projeto pode possuir uma ou mais sessões de teste. Uma sessão pode importar qualquer subconjunto dos mutantes gerados no projeto a qual pertence, informando a porcentagem ou quantidade de mutantes de cada operador ou, então, selecionando o número do mutante ou as linhas que se deseja realizar a mutação. Na Figura 2 é apresentada a página de criação de uma sessão

de teste, na qual são mostrados o menu da ferramenta (área 1), o formulário para a criação de uma sessão (área 2), uma tabela que resume as informações do projeto corrente (área 3) e uma tabela que lista as sessões do projeto corrente (área 4).

Existe também a noção de grupo de sessões que facilita o trabalho cooperativo entre um grupo de testadores. Um grupo é formado por usuários de várias sessões de um mesmo projeto, sendo que os mutantes são particionados entre as sessões do grupo conforme o número máximo de mutantes que cada um pode possuir.

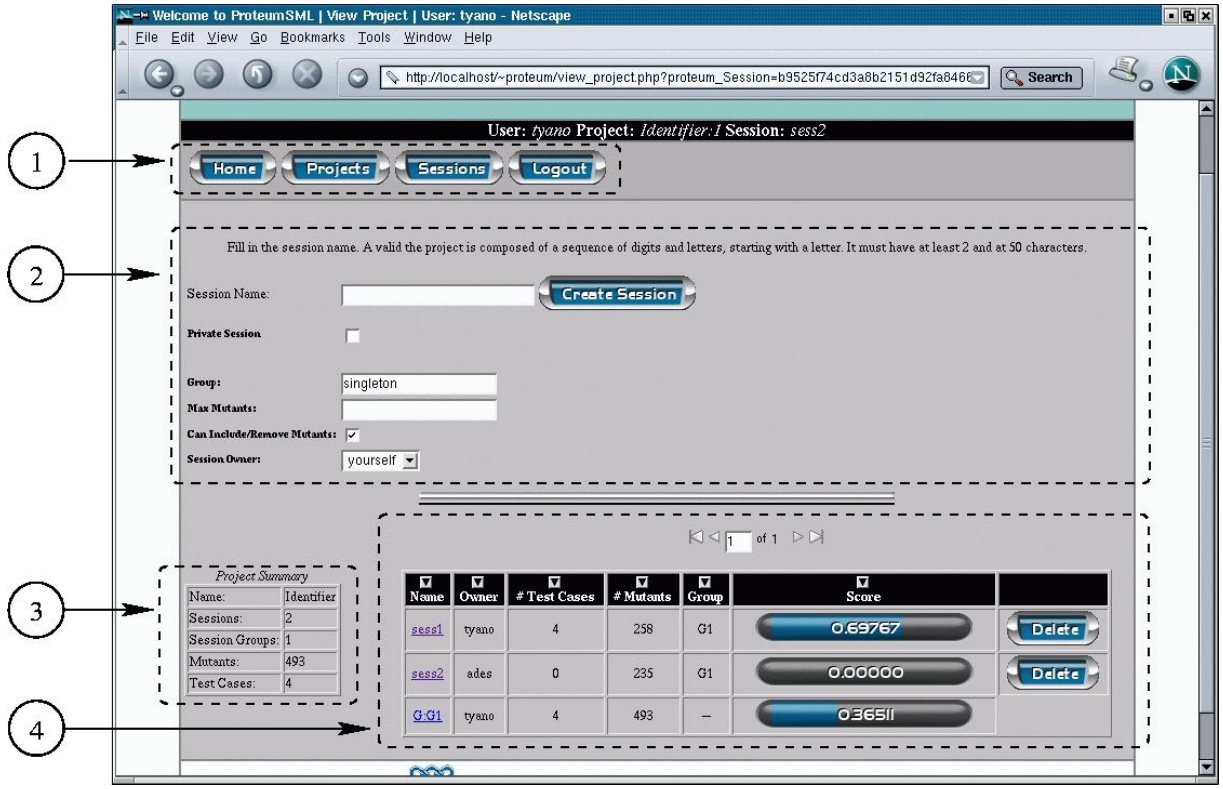


Figura 2: Página de Criação de Sessão

O acesso à ferramenta é controlado através de contas de usuário protegidas por senha. Existem três categorias de usuário: administrador, proprietário de projeto e usuário de sessão. O usuário administrador é criado na instalação da PROTEUM/SML e é responsável pelas autorizações à entrada de outros usuários ao sistema. Um proprietário de projeto tem permissão para criar novos projetos e determina quais usuários podem pertencer ao seu projeto. Já o usuário de sessão tem permissão para criar sessões em um projeto já existente.

Após a criação de uma sessão, pode-se incluir casos de teste tanto interativamente quanto por importação. Em ambos os modos de inclusão, a saída é mostrada ao usuário que deve verificar se está correta, confirmando ou não a inclusão do caso de teste. Na inclusão interativa, o usuário fornece o caso de teste no campo do formulário respectivo. Na importação, o usuário fornece um arquivo ASCII com o caso de teste.

Após a execução dos mutantes com o conjunto de caso de teste, aqueles que permaneceram vivos devem ser analisados para verificar se são equivalentes ou não ao programa em teste. A análise pode ser feita observando-se as linhas do código nas quais a mutação ocorreu. Para tanto, a ferramenta permite a visualização lado a lado do programa em teste e de um mutante, destacando-se as linhas mutadas.

A ferramenta disponibiliza ao testador dois tipos de relatórios estatístico: por operador e por caso de teste. No primeiro, é apresentada a quantidade de mutantes vivos, mortos e equivalentes referentes a cada operador. No segundo, resumem-se informações sobre a execução de cada caso de teste. Tais relatórios permitem realizar uma análise detalhada da adequação dos casos de teste em relação aos mutantes gerados.

Na Figura 3 é apresentada a página que lista os mutantes de uma sessão de teste. Na área 1 encontram-se informações sobre o *status* da sessão, verificando-se principalmente o escore de mutação da sessão que indica a adequação dos casos de teste. Na área 2 podem-se observar os mutantes gerados na sessão.

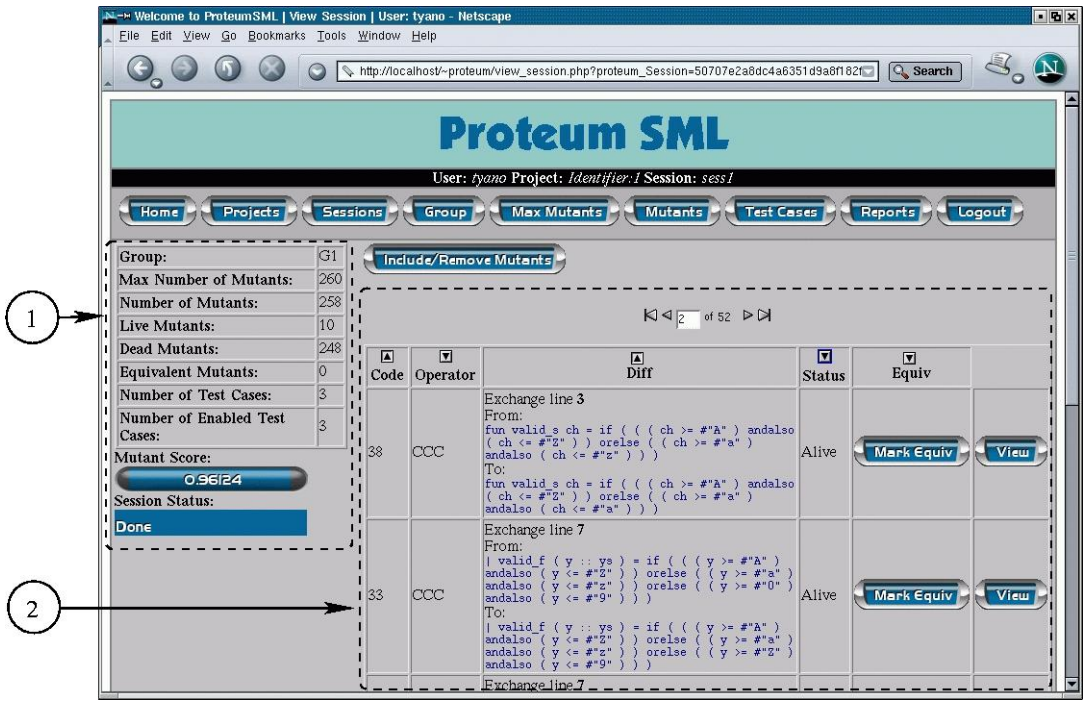


Figura 3: Mutantes de uma Sessão

4.2 Arquitetura e Aspectos Relevantes de Implementação

A arquitetura da PROTEUM/SML é apresentada na Figura 4. As funcionalidades da ferramenta encontram-se divididas entre módulos independentes e dependentes do Teste de Mutação para SML. Os módulos independentes são compostos pelas funcionalidades comuns entre a PROTEUM/SML e a ferramenta PROTEUM/CPN, que está sendo desenvolvida para dar apoio ao Teste de Mutação para Redes de Petri Coloridas [26]. Buscou-se, dessa forma, identificar e isolar as funcionalidades de uma ferramenta de apoio ao Teste de Mutação que sejam independentes da linguagem/especificação alvo, promovendo o reuso de código. Por sua vez, os módulos dependentes possuem as funcionalidades que são estritamente relacionadas à SML. Por exemplo, é necessário realizar um tratamento no código fonte de programas SML para sua execução.

A interação da ferramenta é realizada através de interface *web*. O módulo gerenciador de *skins* é responsável pela formatação das páginas, separando os demais módulos de detalhes específicos de exibição. Dessa forma, pode-se alterar toda a aparência das páginas que compõem a ferramenta sem a necessidade de modificar os códigos referentes às funcionalidades. Além disso, é possível personalizar a interface para se adequar às necessidades de uma linguagem/especificação específica.

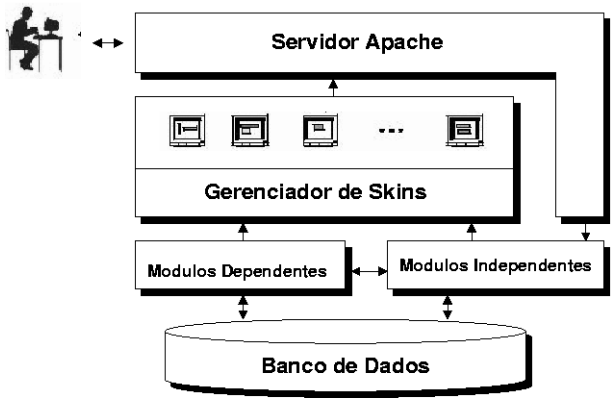


Figura 4: Arquitetura da PROTEUM/SML

Os módulos da PROTEUM/SML foram desenvolvidos na linguagem *PHP 4.1.2*, sobre a plataforma *Linux RedHat 7.3*, kernel versão 2.4.20-18.7, utilizando o *Apache 1.1* como servidor de *WWW* e o *MySQL 3.23.56* como servidor de banco de dados.

Os operadores de mutação, utilizados para a geração dos mutantes na PROTEUM/SML, foram descritos na linguagem $\mathcal{MuDeL}$ (*MUtant DEscription Language*) [27]. A linguagem $\mathcal{MuDeL}$ visa a fornecer precisão e formalidade na descrição dos operadores e a facilitar a automatização da geração de mutantes. Considerando o aspecto crucial dos operadores de mutação no Teste de Mutação, isso é grande importância para se evitar ambigüidades e inconsistências. Foi utilizado o sistema `mudelgen` para "compilar" a descrição dos operadores em $\mathcal{MuDeL}$ e gerar os mutantes, com base na gramática livre de contexto de SML.

O presente trabalho adotou SML'97 e o compilador SML/NJ (*Standard ML of New Jersey*) desenvolvido pelos Laboratórios Bell e pela Universidade de Princeton.

5 Estudo de Caso

Nesta seção é apresentado um estudo de caso da aplicação do Teste de Mutação para SML com o programa exemplo *identifier.sml* (Figura 5), utilizando-se a ferramenta PROTEUM/SML. Esse programa é a implementação em SML do programa *identifier* escrito na linguagem C apresentado em [17]. O programa a ser testado é um verificador de identificadores, que analisa se um determinado identificador é válido. Para ser válido, o identificador deve ser uma cadeia de letras e dígitos, iniciada por uma letra e ter no mínimo 1 e no máximo 6 caracteres.

```
fun no_new_line (#"n"::nil) = nil
| no_new_line (z::zs) = z::no_new_line(zs);

fun valid_s ch =
     if (((ch >= #"A") andalso (ch <= #"Z")) orelse
     ((ch >= #"a") andalso (ch <= #"z")))
     then true
     else false;

fun valid_f nil = true
| valid_f (y::ys) =
     if (((y >= #"A") andalso (y <= #"Z")) orelse
     ((y >= #"a") andalso (y <= #"z")) orelse
     ((y >= #"0") andalso (y <= #"9")))
     then valid_f(ys)
     else false;

fun valid_all nil = false
| valid_all (x::xs) =
     if valid_s(x)
     then valid_f(xs)
     else false;

fun ident s =
     let
          val value_l = no_new_line(explode(s));
          val length_l = length(value_l)
     in
          if (valid_all(value_l) andalso length_l >= 1 andalso length_l <= 6)
          then "Valid\n"
          else "Invalid\n"
     end;

print("Input a identifier\n");
val value = TextIO.inputLine(infile);
val result = ident(value);
print(result);
```

Figura 5: Programa *identifier.sml*

Na Tabela 2 são apresentadas algumas informações sobre o programa *identifier.sml*, tais como o número de linhas de código, de variáveis, de constantes, de operadores e de expressões, as quais influenciam na quantidade de mutantes gerados no Teste de Mutação desse programa.

Tabela 2: Informações do Programa *identifier.sml*

Descrição	*Qtde*
Linhas de Código	25
Variáveis	12
Constantes	12
Operadores	21
Expressões	17

Geração de Mutantes

A quantidade de mutantes gerados por cada operador na PROTEUM/SML para o programa exemplo é apresentada na Tabela 3. A aplicação de todos os operadores gera 493 mutantes, sendo 248, 104, 10, 131 pelos operadores de mutação de constantes, operadores, expressões e variáveis, respectivamente.

Tabela 3: Mutantes do Programa *identifier.sml*

Operador	*Qtde*	*Operador*	*Qtde*
CBR	6	EDL	2
CCC	80	EME	0
CCS	115	ETI	8
CRC	47	VAV	4
OAR	0	VDT	0
OLC	4	VRF	0
OLN	30	VSV	127
OLR	10	VTM	0
ORR	60		
Quantidade Total: 493			

Criação de Casos de Teste

Foi construído um conjunto T de casos de teste adequado aos mutantes do programa *identifier.sml* gerados por todos os operadores de mutação da PROTEUM/SML. Um caso de teste é denotado pelo par de entrada e e a saída s: (e, s).

Inicialmente, foi inserido o conjunto de casos de teste $T_0 = \{(a1, válido), (2B3, inválido), (Z - 12, inválido), (A1b2C3d, inválido)\}$. Após a execução dos mutantes com T_0, o escore de mutação obtido é 0.77282. A fim de melhorar o escore de mutação foi adicionado o conjunto de casos de teste $T_1 = \{(zzz, válido), (aA, válido), (A1234, válido), (aa09, válido)\}$, obtendo escore igual a 0.88641. Mesmo com a inclusão de T_0 e T_1, o escore de mutação ainda não é satisfatório. Assim, foi elaborado o conjunto de casos de teste $T_2 = \{(1\#, inválido), (AAA, válido), ([, inválido), (\{, inválido), (x :, inválido), (x[[, inválido), (x\{\{, inválido), (@, inválido), (xy\_, inválido)\}$, melhorando o escore para 0.97363. A partir de então, foram identificados quatro mutantes equivalentes. Esses mutantes permaneceram vivos mesmo após a inclusão de T_2 e, ao analisá-los, foi observado que eles teriam sempre o mesmo comportamento do programa original, assim nenhum caso de teste poderia matá-los. Ao incluir o conjunto de caso de teste $T_3 = \{(c, válido), (ZZZZ, válido)\}$ e $T_4 = \{(ABCEDF, válido)\}$ foi obtido escore de mutação igual a um e o conjunto de casos de teste $T = T_0 \cup T_1 \cup T_2 \cup T_3 \cup T_4$ constitui um conjunto 100% adequado ao Teste de Mutaçao para o programa *identifier.sml*.

Na Tabela 4 mostra-se a evolução da atividade de teste no programa *identifier.sml* com os conjuntos de casos de teste que compõem T. São mostrados os valores do escore de mutação a cada inclusão de T_i.

Tabela 4: Evolução do Escore de Mutação do Programa *identifier.sml*

Conj. de casos de teste	*Escore de mutação*
T_0	0.77282
$T_0 \cup T_1$	0.88641
$T_0 \cup T_1 \cup T_2$	0.97363
$T_0 \cup T_1 \cup T_2 \cup T_3$	0.98986
$T_0 \cup T_1 \cup T_2 \cup T_3 \cup T_4$	1.0000

6 Conclusão

Atualmente, há poucos trabalhos relacionados ao teste de programas funcionais, sendo que os existentes não possuem uma medida de cobertura da atividade de teste. Assim, neste trabalho, buscou-se estabelecer subsídios para a investigação da adequabilidade da aplicação de critérios de teste tradicionalmente utilizados em programas imperativos no teste de programas funcionais. Particularmente, investigou-se a aplicabilidade do critério Teste de Mutação para o teste de programas em SML (*Standard Meta Language*). O Teste de Mutação é um critério de teste que fornece uma maneira de auxiliar na geração e na avaliação de um conjunto de casos de teste, sendo explorado em diversos contextos, como no teste de unidade, de integração e de especificação.

Um dos principais aspectos para caracterizar o Teste de Mutação é a definição de um conjunto de operadores de mutação específico para a linguagem alvo. A escolha dos operadores determina tanto os tipos de erros que se deseja revelar e a cobertura que se quer garantir na atividade de teste, como também determina a eficiência da utilização do Teste de Mutação, uma vez que o custo computacional está relacionado ao número de mutantes gerados pela aplicação dos operadores de mutação. Neste trabalho, foi apresentado um conjunto de 17 operadores de mutação para a linguagem SML definido em [30]. Os operadores de mutação foram descritos na linguagem $\mathcal{MuDeL}$, a qual fornece precisão e formalidade na descrição dos mesmos como também facilita a automatização da geração de mutantes.

A disponibilidade de ferramentas de apoio ao Teste de Mutação contribui para aumentar a produtividade, visto que a quantidade de informações que deve ser tratada é muito grande. Para dar apoio à aplicação do Teste de Mutação para SML, foi desenvolvida a ferramenta *web* PROTEUM/SML. Sua implementação teve como base a ferramenta PROTEUM/CPN [26], a qual oferece apoio ao Teste de Mutação de Redes de Petri Coloridas.

Com o objetivo de avaliar a eficácia dos operadores de mutação definidos para SML, deve-se realizar estudos empíricos com exemplos que abrangem as principais características da linguagem. A fim de complementar o conjunto de operadores de mutação para SML, pode-se investigar operadores que explorem as características de encapsulamento, polimorfismo e modularidade de SML. Além disso, como foram definidos apenas operadores para o teste de unidade, pode-se investigar operadores para o teste de integração.

Outro trabalho que pode ser realizado é aplicar o Teste de Mutação de SML no código da especificação em Redes de Petri Coloridas produzida pela ferramenta PROTEUM/CPN, que também é escrito em SML. O objetivo seria a investigação da relação entre o teste de especificações de sistemas e das respectivas implementações desses sistemas, com enfâse no Teste de Mutação, utilizando-se as ferramentas PROTEUM/SML e PROTEUM/CPN. Na essência o propósito desse trabalho, seria buscar o relacionamento entre os conceitos de teste aplicados no nível de especificação e os do nível de implementação. No caso particular do Teste de Mutação, buscar-se-ia o relacionamento entre os operadores de mutação em nível de especificação e os operadores de mutação em nível de implementação.

Além disso, com a implementação da PROTEUM/SML viabiliza-se a realização de estudos empíricos que visem à investigação do relacionamento dos operadores de mutação definidos para SML com os de C. O propósito desse estudo seria buscar a relação dos conceitos de teste aplicados na programação funcional e imperativa.

Com base na experiência no desenvolvimento da PROTEUM/SML, juntamente com da ferramenta PROTEUM/CPN, motiva-se a abstração de um *framework* para o desenvolvimento de ferramentas de apoio ao Teste de Mutação, visto que foram identificadas as funcionalidades indenpendentes à linguagem/especificação alvo desse critério.

Referências

[1] Agrawal, H. (1989). Design of mutant operators for the C programming language. Relatório Técnico SERC-TR-41-P, Software Engineering Research Center/Purdue University.

[2] Bieman, J. M., Ghosh, S., e Alexander, R. T. (2001). A technique for mutation of Java objects. In *16th IEEE Internation Conference on Automated Software Engineering*, pp. 23–26, San Diego, CA.

[3] Budd, A. T. (1981). *Mutation Analysis: Ideas, Examples, Problems and Prospects*, cap. Computer Program Testing, pp. 129–148. North-Holland Publishing Company.

[4] Claessen, K. e Hughes, J. (2000). Quickcheck: a lightweight tool for random testing of haskell programs. In *Fifth ACM SIGPLAN international conference on Functional programming*, pp. 268–279. ACM Press.

[5] Claessen, K. e Hughes, J. (2002). Testing monadic code with quickcheck. *SIGPLAN Not.*, 37(12):47–59.

[6] Claessen, K., Runciman, C., Chitil, O., Hughes, J., e Wallace, M. (2002). Testing and tracing lazy functional programs. In 4^{th} *Summer School in Advanced Functional Programming*, Oxford.

[7] Delamaro, M. E. (1993). Proteum – Um ambiente de teste baseado na análise de mutantes. Dissertação de mestrado, ICMC/USP, São Carlos, SP.

[8] Delamaro, M. E., Maldonado, J. C., e Mathur, A. P. (2001). Interface mutation: An approach for integration testing. *IEEE Transactions on Software Engineering*, 27(3):228–247.

[9] DeMillo, R. A., Lipton, R. J., e Sayward, F. G. (1978). Hints on test data selection: Help for the practicing programmer. *IEEE Computer*, 11(4):34–41.

[10] DeMillo, R. A. e Offutt, A. J. (1991). Constraint-based automatic test data generation. *IEEE Transactions on Software Engineering*, 17(9):900–910.

[11] Fabbri, S. C. P. F., Maldonado, J. C., Delamaro, M. E., e Masiero, P. C. (1999a). Proteum/FSM: A tool to support finite state machine validation based on mutation testing. In *XIX SCCC - International Conference of the Chilean Computer Science Society*, pp. 96–104, Talca, Chile.

[12] Fabbri, S. C. P. F., Maldonado, J. C., Sugeta, T., e Masiero, P. C. (1999b). Mutation testing applied to validate specifications based on statecharts. In *10^{th} International Symposium on Software Reliability Engineering (ISSRE'99)*, pp. 210–219, Boca Raton, Flórida, EUA.

[13] Herington, D. (2002). *HUnit 1.0 User's Guide*.

[14] Kim, S., Clark, J. A., e Mcdermid, J. A. (2000). Class mutation: Mutation testing for object-oriented programs. In *Object-Oriented Software Systems – OOSS*.

[15] Kim, S., Clark, J. A., e Mcdermid, J. A. (2001). Investigating the effectiveness of object-oriented testing strategies with the mutation method. *Software Testing, Verification and Reliability*, 11(4).

[16] Ma, Y.-S., Kwon, Y.-R., e Offutt, J. (2002). Inter-class mutation operators for Java. In *13th International Symposium on Software Reliability Engineering - ISSRE'2002*, pp. 352–366, Annapolis, MD. IEEE Computer Society Press.

[17] Maldonado, J. C., Barbosa, E. F., Vincenzi, A. M. R., Delamaro, M. E., Souza, S. R. S., e Jino, M. (2003). Introdução ao teste de software. In *XVII SBES – Simpósio Brasileiro de Engenharia de Software (Minicurso)*, Manaus, AM.

[18] Milner, R., Tofte, M., e Harper, R. (1990). *The Definition of Standard ML*. The MIT Press, Cambridge, Mass.

[19] Milner, R., Tofte, M., Harper, R., e MacQueen, D. (1997). *The Definition of Standard ML (Revised)*. The MIT Press, Cambridge, Mass.

[20] Moss, G. E. e Runciman, C. (1999). Automated benchmarking of functional data structures. *Lecture Notes in Computer Science*, 1551:1–15.

[21] Oliveira, K. A., Machado, P. D. L., e Andrade, W. L. (2003). CaslTest - test case, test oracle and test data generation from Casl specifications. In *Sessão de Ferramentas do XVII SBES – Simpósio Brasileiro de Engenharia de Software*, pp. 73–78, Manaus, AM.

[22] Probert, R. L. e Guo, F. (1991). Mutation testing of protocols: Principles and preliminary experimental results. In *IFIP TC6 Third International Workshop on Protocol Test Systems*, pp. 57–76, North-Holland.

[23] Pucella, R. (2001). Notes on programming Standard ML of New Jersey. Relatório Técnico Version 110.0.6, Departament of Computer Science, Cornell University, Ithaca, NY.

[24] Ritchey, R. W. (2000). Mutation network models to generate network security test cases. In *Mutation 2000 Symposium*, pp. 101–108, San Jose, California.

[25] Simão, A. S., Maldonado, J. C., e Fabbri, S. C. P. F. (2000). Proteum-RS/PN: A tool to support edition, simulation and validation of Petri nets based on mutation testing. In *XIV SBES – Simpósio Brasileiro de Engenharia de Software*, pp. 227–242, João Pessoa, PB.

[26] Simão, A. S. (2002). *Teste e Validação de Redes de Petri Coloridas Usando Análise de Mutantes*. Qualificação de doutorado, ICMC/USP, São Carlos, SP.

[27] Simão, A. S. e Maldonado, J. C. (2002). MuDeL: A language and a system for describing and generating mutants. *Journal of the Brazilian Computer Society*, 8(1):73–86.

[28] Souza, S. R. S., Maldonado, J. C., Fabbri, S. C. P. F., e Lopes de Souza, W. (2000). Mutation testing applied to Estelle specifications. In *33^{rd} Hawaii Internacional Conference on System Sciences, Mini-Tracks: Distributed Systems Testing*, Maui, Havaii.

[29] Ullman, J. D. (1998). *Elements of ML Programming - ML97 Edition*. Prentice Hall, New Jersey, USA.

[30] Yano, T. (2004). Estudo do teste de mutação em programas funcionais sml. Dissertação de mestrado, ICMC/USP, São Carlos, SP.

[31] Yano, T., Simao, A. S., e Maldonado, J. C. (2003). Proteum/SML: Uma ferramenta de apoio ao teste de mutação para a linguagem Standard ML. In *Sessão de Ferramentas do XVII SBES – Simpósio Brasileiro de Engenharia de Software*, pp. 67–72, Manaus, AM, Brasil.

Estudio del Espacio de Soluciones del Problema del Cajero Viajante

Osvaldo Gómez
Universidad Nacional de Asunción
Centro Nacional de Computación
San Lorenzo, Paraguay
ogomez@cnc.una.py

Pedro Esteban Gardel Sotomayor
Universidad Nacional de Asunción
Centro Nacional de Computación
San Lorenzo, Paraguay
pgardel@cnc.una.py

Benjamín Barán
Universidad Nacional de Asunción
Centro Nacional de Computación
San Lorenzo, Paraguay
bbaran@cnc.una.py

Resumen

El presente trabajo estudia el espacio de soluciones del *Traveling Salesman Problem (TSP*). Debido al enorme tamaño del espacio de soluciones de los problemas estudiados, se ha decidido tomar muestras de los mismos con el objetivo de tener una visión general de su estructura. Para este fin, se utilizaron dos políticas en la obtención de las muestras, una basada en estudios anteriores y otra aquí propuesta por los autores. Se han tomado veinte instancias del *TSP* de la *TSPLIB*. El análisis de los resultados obtenidos es congruente en todos los casos con la conjetura de un espacio de soluciones globalmente convexo del *TSP*. Esto es, un espacio con características de "*Gran Valle*" como fuera sugerido por Boese, y verificado experimentalmente en el presente trabajo.

Palabras Claves: *Traveling Salesman Problem*, Optimización local, Espacio globalmente convexo.

Abstract

The present paper studies the space of solutions of the Traveling Salesman Problem (TSP). Due to the enormous size of the solutions space of the studied problems, it has been decided to take samples with the objective of having a general vision of the problem structure. For this purpose, two policies were used to obtain the samples, one already used in previous studies and the other designed by the authors. For this study, 20 instances of the *TSP* have been taken from the *TSPLIB*. The analysis of the results was coherent with the conjecture of a globally convex structure of the *TSP*'s solutions space. That is, a space that has characteristics of "*Great Valley*" as it was suggested by Boese and experimentally proved in the present work.

Keywords: Traveling Salesman Problem, Local optimization, Globally convex space.

1 Introducción

El problema del cajero viajante, más conocido por sus siglas en inglés *TSP (Traveling Salesman Problem)*, ha sido utilizado como "*Problema de Prueba*" (o problema *paradigma*) para estudiar el comportamiento de muchos algoritmos de optimización [1,2,4,5] por lo que resulta interesante un estudio más extenso del espacio de soluciones del mismo con el fin de conocer su estructura. Este trabajo se inspira en las propuestas de Boese [1,2] y otros autores como Stützle et al. [5] que han realizado algunos estudios similares para instancias específicas del *TSP*, pero no de manera sistemática, como se propone en el presente trabajo. Estos estudios consistieron en general en hallar óptimos locales mediante distintos algoritmos de búsqueda local para luego analizar los resultados obtenidos e intentar entender la estructura del *TSP* para las instancias estudiadas.
En contrapartida a los trabajos citados, el presente trabajo analiza un conjunto de 20 problemas de solución óptima conocida, extraídos de la *TSPLIB* [6], una fuente de problemas referenciales del *TSP*, muy utilizada por diversos investigadores del área. El presente trabajo propone entonces obtener soluciones factibles del espacio de búsqueda (o muestras), compararlas con la solución óptima y con las demás soluciones factibles, para finalmente intentar conocer la estructura del *TSP*. Para obtener estas muestras, se utilizaron básicamente 2 políticas de muestreo:

1. Siguiendo los trabajos de Boese [1,2], se utilizó una heurística de optimización local, conocida como *3-opt*, con iniciación aleatoria.
2. Alternativamente, se generaron muestras de soluciones uniformemente distribuidas en el espacio de soluciones, para cada problema en estudio.

El trabajo fue organizado de la siguiente manera: a continuación se describe el *TSP* y se formaliza la notación a ser utilizada. Seguidamente, se describen los métodos utilizados para obtener las muestras de soluciones del *TSP*. En la cuarta sección se presentan los resultados experimentales, mientras que en la sección 5 se describe la estructura del *TSP* como un espacio globalmente convexo. Finalmente se presentan las conclusiones del estudio.

2 Problema del Cajero Viajante.

El *TSP* puede ser representado por un conjunto de ciudades o nodos $V=\{N1,N2,\ldots..Nn\}$ con n elementos (o ciudades) y un conjunto A de arcos que asocia a cada par de ciudades o nodos $\{Ni, Nj\}$ una distancia $d(i,j)$. La solución del *TSP* es conocida como *tour* (r) y está dada por un conjunto ordenado de las n ciudades o nodos $\{Ni\}$ que contiene a todas las ciudades del problema exactamente una vez, esto es:

$$r = \{N(r(1)), N(r(2)), ..., N(r(n))\}; \qquad N(r(i)) \in V, \quad N(r(i)) \neq N(r(j)) \text{ si } i \neq j \qquad (1)$$

Cada *tour* tiene asociado una longitud definida como:

$$l(r) = \sum_{i=1}^{n-1} d(N(r(i)), N(r(i+1))) + d(N(r(n), N(r(1))). \qquad (2)$$

El objetivo del *TSP* es hallar al *tour* de menor longitud, también conocido como *tour* óptimo (r^*).

El presente trabajo sólo considera problemas en los que los arcos representen distancias euclidianas entre los nodos i y j, por lo tanto, $d(i,j) = d(j,i)$. Este tipo de problemas es conocido como *TSP simétrico*. En este contexto, dados dos *tours* $r1$ y $r2$, la distancia $\delta(r1, r2)$ se define como n menos la cantidad de arcos comunes a ambos *tours*. Lógicamente, la distancia de un *tour* r a la solución óptima r^* estará denotada como $\delta(r, r^*)$.

Una población P es un conjunto de m soluciones o *tours*, esto es:

$$\text{Población:} \quad P = \{r1, r2, \ldots, rm\} \qquad (3)$$

La distancia media de un *tour* r a una población P se define como

$$\delta(P, r) = \frac{1}{m} \sum_{i=1}^{m} \delta(ri, r) \qquad (4)$$

y da una idea de cuan cerca esta un *tour* a una población P. Corresponde hacer notar que dada una población P uniformemente distribuida en un sub-espacio, los *tours* que se encuentren en su centro tendrán una menor distancia media a la población que los que se encuentren en su periferia, por lo que $\delta(P,r)$ podrá ser utilizado para estimar la posición de un punto respecto a una población.

3 Métodos de muestreo utilizados para el análisis

Inspirado en los trabajos de Boese [1,2], el primer método utilizado intenta crear una población de soluciones localmente óptimas, para lo que se utilizará un algoritmo que se dará en llamar "*3-opt*". El mismo utiliza una heurística de búsqueda local mostrada en el *Pseudocódigo 1*, para hallar m óptimos locales y conformar así una población de estudio P. Este acercamiento ya fue abordado por otros autores en [1,3,5] demostrándose que las

soluciones obtenidas se concentran en una zona determinada del espacio de soluciones y no se distribuyen uniformemente en todo el espacio.
A raíz de la distribución no uniforme de la población de óptimos locales encontrada en publicaciones anteriores, este trabajo propone adicionalmente el estudio de una población más uniforme, con soluciones distribuidas a lo largo de un mayor espacio de soluciones, para lo cual se propone un segundo método de selección de muestras. Este método, que damos en llamar "*Dist*", obtiene un conjunto de soluciones con la característica de tener un subconjunto de cardinalidad constante, perteneciente a cada posible distancia del óptimo r^*.

3.1 Descripción del algoritmo *3-opt*

Al inicio, esta heurística guarda en una variable r_{opt} un *tour* hallado aleatoriamente de la siguiente manera: el *tour* comienza en una ciudad cualquiera del problema, luego se elige aleatoriamente otra ciudad de *Vc*, siendo *Vc* el conjunto de ciudades todavía no visitas, y se elimina ésta de *Vc*. El proceso se repite iterativamente hasta que el conjunto *Vc* quede vacío y se hayan visitado todas las ciudades del problema, por lo que solo queda regresar a la ciudad de origen.
Una vez encontrado un *tour* inicial, se realiza iterativamente el siguiente proceso: Se cambian tres arcos elegidos al azar de r_{opt}, dando origen a un nuevo *tour* válido que se guarda en r_{new}. Se comparan las longitudes de r_{opt} y r_{new}, si $l(r_{new})$ es menor, se guarda r_{new} en r_{opt}. Se realiza este proceso hasta que r_{opt} no varíe durante una cantidad de iteraciones (*CI*) igual a 1000 para los resultados experimentales presentados en este trabajo.
El proceso mencionado se repite hasta obtener el Número de Tours (*NT*) deseado para generar la población *P*. Para este trabajo, se escogió *NT* = 1000 tours y se calcularon los correspondientes valores de longitud $l(r)$, distancia al óptimo $\delta(r, r^*)$ y distancia media a la población $\delta(P,r)$, para cada tour $r \in P$.

Pseudocódigo 1: *3-opt*

Parámetros de entrada: matriz de distancias $D = d(i,j)$, número de tours *NT*, cantidad de iteraciones *CI*
Parámetros de salida: población *P*, con los correspondiente valores de $l(r)$, $\delta(r, r^*)$ y $\delta(P,r)$
Inicializar $k = 0$
Repetir mientras $k < NT$
  r_{opt} = *Nuevo tour ( )*
  Hallar $l(r_{opt})$
  Inicializar $c = 0$
  Repetir mientras $c < CI$
    r_{new} = *Cambios* (r_{opt})
    Hallar $l(r_{new})$
    Si $l(r_{new}) < l(r_{opt})$
      $r_{opt} = r_{new}$
      $l(r_{opt}) = l(r_{new})$
      c=0
    Fin si
    c=c+1
  Fin mientras
P(k) = r_{opt}
Calcular $\delta(r_k, r^*)$
Calcular $l(r_k)$
$k = k + 1$
Fin mientras
Calcular $\delta(P,r)$

Función r_{opt} = *Nuevo tour ( )*
r_{opt} = conjunto vacío
Mientras *Vc* no sea vacío
  Se escoge al azar una ciudad (*Ni*) de *Vc*
  Se elimina *Ni* de *Vc*
  r_{opt} = (r_{opt}, *Ni*)
Fin mientras

Función r_{new} = *Cambios* (r_{opt})
Seleccionar tres arcos al azar de r_{opt}
Cortar r_{opt} en los arcos seleccionados
Reordenar los segmentos de r_{opt} para construir un nuevo tour (r_{new}) con $\delta(r_{new}, r_{opt}) = 3$

3.2 Descripción del Algoritmo *Dist*

Este algoritmo tiene como objetivo hallar un numero de *tours* (*NT'*) a todas las distancias posibles del óptimo *r**. Las distancias posibles al óptimo van de 2 hasta *n*, siendo *n* la cantidad de ciudades en el problema. Para cada una de estas distancias se genera entonces un subconjunto de *NT'* soluciones, por lo que el número total de tours estudiados es de: *NT'(n*-2). Por ejemplo, para un problema de 280 ciudades (como el *a280* mostrado en la tabla 1), con *NT'*=24, se analizaron 6672 tours componentes de la población *P*, mientras que para el problema *pr1002* con *NT'=6* (ver Tabla 1) la población *P* contenía solo 6000 tours.
Este algoritmo selecciona arcos al azar en el *tour* óptimo *r** y los reordena formando nuevo *tour* válido r_{new}. Seguidamente se calcula $\delta(r^*, r_{new})$ y se comprueba si r_{new} está a la distancia deseada, de no ser así, se continua el proceso de cambio de arcos. Una vez que se han obtenido la cantidad *NT'* de *tours* deseados a una distancia dada del óptimo, se procede a hallar las siguientes muestras, posiblemente a la distancia siguiente. En consecuencia, el algoritmo también produce una población *P* de *tours*, con sus correspondientes valores de $l(r)$, $\delta(r, r^*)$ y $\delta(P,r)$. A continuación se describe el pseudocódigo correspondiente.

Pseudocódigo 2: *Dist*

Parámetros de entrada: *r*, NT'*
Parámetros de salida: población *P*, con los correspondiente valores de $l(r)$, $\delta(r, r^*)$ y $\delta(P,r)$
NC = cantidad de ciudades en *r**
Inicializar *P* como conjuntos vacíos
Para *nc* = 2 hasta *NC*
  Para *k* = 1 hasta *NT'*
    r_{new} = *Cambios (r*, nc)*
    Hallar $l(r_{new})$
    $P = [P, r_{new}]$
    $l(r_k) = l(r_{new})$
    Calcular $\delta(r_k, r^*)$]
  Fin para
Fin para
Calcular $\delta(P,r)$

Función *Cambios (r*, nc)*
$r_{new} = r^*$
Mientras $nc > \delta(r_{new}, r^*)$
  Seleccionar arcos al azar de r_{new} y cortar el *tour*
  Reconstruir r_{new} con los segmentos y obtener un nuevo r_{new}
  Hallar $\delta(r_{new}, r^*)$
  Si $\delta(r_{new}, r^*)$ tiene distancia buscada, retornar r_{new}
Fin mientras.

4 Análisis de la estructura del espacio de soluciones del TSP

4.1 Problemas Estudiados

Se utilizaron un total de 20 problemas con soluciones óptimas ya conocidas. Estos problemas ampliamente difundidos [6], se detallan a continuación en la Tabla 1 que muestra el nombre del problema, el número *n* de ciudades, los parámetros *NT* y *NT'*, así como la cantidad de tours analizados.

La cantidad *NT'* = 24 fue elegida para permitir obtener una cantidad apreciable de soluciones en los problemas pequeños. En los problemas de mayor tamaño se disminuyó *NT'* debido al enorme tiempo de procesamiento que requiere poblaciones muy grandes.

Por falta de suficientes recursos computacionales, no se realizó la búsqueda de soluciones con *3-opt* en los problemas de más de 101 ciudades, lo que no invalida las conclusiones del trabajo, dada la uniformidad de los resultados experimentales que serán mostrados en la siguiente sección.

Nombre del Problema en TSPLIB	Número de ciudades (*n*)	*Tours* en cada subconjunto (*NT'*) (usando *Dist*)	Tamaño de la Población *P* (usando *Dist*)	Cantidad de *tours NT* (usando *3-opt*)
fri26	26	24	600	1000
bayg29	29	24	672	1000
bays29	29	24	672	1000
att48	48	24	1128	1000
eil51	51	24	1200	1000
berlin52	52	24	1224	1000
st70	70	24	1656	1000
pr76	76	24	1800	1000
eil76	76	24	1800	1000
kroa100	100	24	2376	1000
kroc100	100	24	2376	1000
krod100	100	24	2376	1000
eil101	101	24	2400	1000
lin105	105	24	2496	
ch130	130	24	3096	
ch150	150	24	3576	
tsp225	225	24	5376	
a280	280	24	6696	
pcb442	442	9	3969	
pr1002	1002	6	6006	

Tabla 1: Resumen de problemas en estudio.

4.2 Presentación de los resultados

En las poblaciones halladas con el algoritmo *Dist* no se consideraron individuos repetidos, es decir, las poblaciones solo contenían una vez a cada solución encontrada. En cambio, en las poblaciones halladas con la heurística *3-opt* se hallaron 1000 individuos diferentes en la mayoría de los problemas, excepto en los siguientes casos:

- En *fri26*, donde se obtuvieron 687 individuos diferentes y se alcanzo el óptimo en 87 ocasiones.
- En *bayg29*, donde se encontraron 877 individuos y se llego al óptimo 57 veces, y
- en *bays29*, donde se obtuvieron 897 *tours* no repetidos y se encontró el óptimo en 35 ocasiones.

Cabe destacar que en ninguno de los demás problemas considerados se halló el óptimo. Para cada población *P* se calcularon los siguientes valores de correlación, mostrados en la Tabla 2:

- La correlación entre la distancia al óptimo $\delta(r, r^*)$ y la longitud del *tour l(r)*, denotado como $\rho(\delta(r, r^*); l(r))$.
- La correlación entre la distancia media a la población $\delta(P,r)$ y la longitud de los *tours l(r)* denotado como $\rho(\delta(P,r); l(r))$.
- La correlación entre la distancia al óptimo $\delta(r, r^*)$ y la distancia media a la población $\delta(P,r)$ denotado como $\rho(\delta(r, r^*); \delta(P,r))$.

Como puede observase en la Tabla 2, las correlaciones calculadas son muy significativas en todos los casos, siendo en general más altas para las poblaciones uniformes generadas con el algoritmo *Dist* que para las poblaciones de óptimos locales generados con la heurística *3-opt*. Cabe notar además que el valor de estas correlaciones crece en general con el tamaño *n* del problema, por lo que se puede presumir que en general, estas correlaciones aumentan con la complejidad del problema, por lo que resulta razonable intentar analizar estos resultados experimentales para intentar conocer la estructura del *TSP*, especialmente para el caso de problemas de creciente complejidad.

Antes de iniciar el referido estudio, se presentan las figuras 1 al 8, ilustrando para algunos de los problemas tipo estudiados las características resaltantes de las 3 variables estudiadas. Debido a la enorme cantidad de problemas estudiados, resulta imposible presentar figuras para cada problema analizado. Seguidamente se presentan los gráficos de los problemas *fri26*, *berlin52*, *kroa100*, *pcb442* y *pr1002*.
Para cada problema se muestran tres gráficos. En las figuras 1, 2 y 3 se presentan los resultados usando la heurística *3-opt* y en las figuras 4 al 8 los resultados usando el algoritmo *Dist*.

Correlaciones Experimentales para poblaciones halladas con el algoritmo *Dist*			
	$\rho(\delta(r,r^*); l(r))$	$\rho(\delta(P,r); l(r))$	$\rho(\delta(r,r^*); \delta(P,r))$
fri26	0.95448091033952	0.95619980040881	0.99968988875798
bayg29	0.96133911986948	0.96009781141200	0.99979852774148
bays29	0.96070250670296	0.95803519904316	0.99984148510148
att48	0.97170200570656	0.97164083616104	0.99996217348502
eil51	0.97995087068297	0.97993644346292	0.99997253792214
berlin52	0.97834824516068	0.97817268025183	0.99996432970952
st70	0.98645264864984	0.98642983813238	0.99998855603249
pr76	0.98678670617501	0.98680411134174	0.99999198845579
eil76	0.98703033452627	0.98703837221474	0.99999132039835
kroa100	0.98807255595606	0.98806085294992	0.99999613193151
kroc100	0.98799045322066	0.98799244346144	0.99999496117053
krod100	0.98889865432748	0.98886823772830	0.99999600926837
eil101	0.99012944939611	0.99011129306537	0.99999623319082
lin105	0.98824266471120	0.98823017753611	0.99999675465118
ch130	0.99274096381607	0.99275294041509	0.99999754184527
ch150	0.99428751931525	0.99428768016066	0.99999847528659
tsp225	0.99561774098757	0.99561979503179	0.99999931367740
a280	0.99645694386082	0.99645797223804	0.99999954320402
pcb442	0.99816600999945	0.99816520843327	0.99999958515832
pr1002	0.99913041235388	0.99913038639726	0.99999989057276
Correlaciones Experimentales para poblaciones halladas con heurística *3-opt*			
	$\rho(\delta(r,r^*); l(r))$	$\rho(\delta(P,r); l(r))$	$\rho(\delta(r,r^*); \delta(P,r))$
fri26	0.79740539145696	0.82356744995048	0.78618554174895
bayg29	0.75737847421895	0.87465290225778	0.88584646493907
bays29	0.69656272456600	0.86772165118454	0.77833781021983
att48	0.59475322499705	0.82072366195982	0.75013111753384
eil51	0.65725125076772	0.91529026136113	0.75356737848526
berlin52	0.71519426726462	0.86492285540173	0.84911820026034
st70	0.60717715417841	0.89953192544241	0.67846357101462
pr76	0.70820045789984	0.88077517172286	0.83383889710767
eil76	0.69559421901902	0.94123566483957	0.77263029989318
kroa100	0.69158036378025	0.90389926628806	0.84802935161287
kroc100	0.69957464878138	0.92368211552670	0.79578866223248
krod100	0.69873339821833	0.91241888905004	0.79223990205942
eil101	0.70788007788079	0.94446148831836	0.77358426643261

Tabla 2: Resumen de los resultados experimentales.

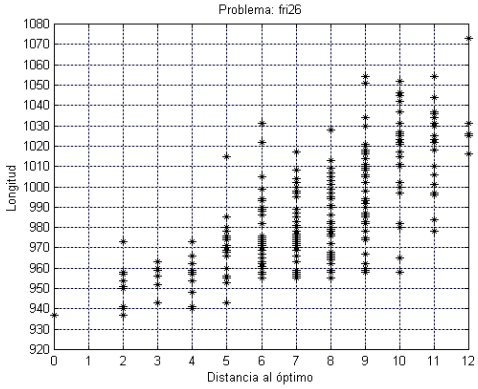


Figura 1.1: *l(r)* en función de *δ(r, r*)*

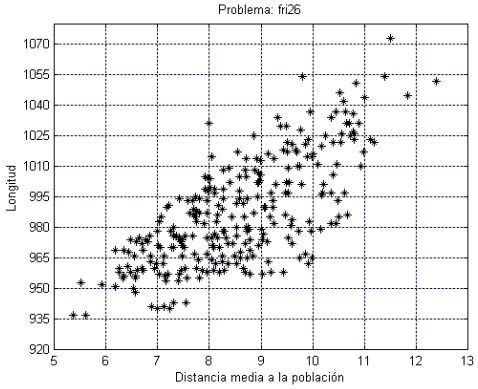


Figura 1.2: *l(r)* en función de *δ(P, r*)*

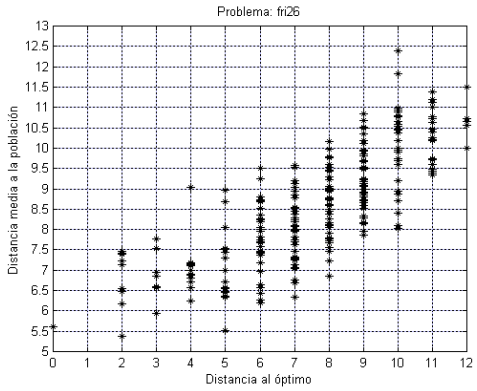


Figura 1.3: *δ(P, r*)* en función de *δ(r, r*)*

Figura 1: Problema *fri26* usando *3-opt*

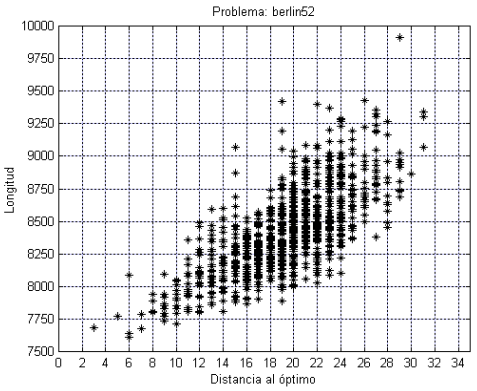


Figura 2.1: *l(r)* en función de *δ(r, r*)*

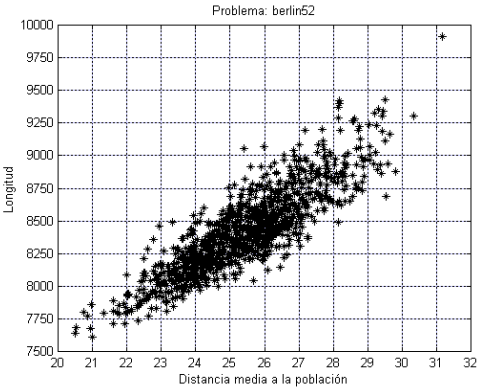


Figura 2.2 *l(r)* en función de *δ(P, r*)*

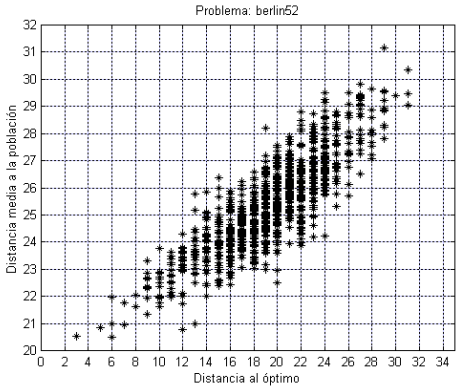


Figura 2.3: *δ(P, r*)* en función de *δ(r, r*)*

Figura 2: Problema *berlin52* usando *3-opt*

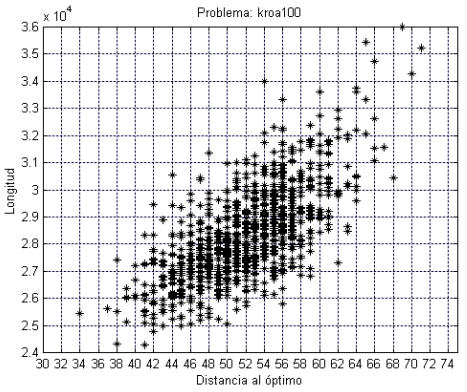


Figura 3.1: *l(r)* en función de *δ(r, r*)*

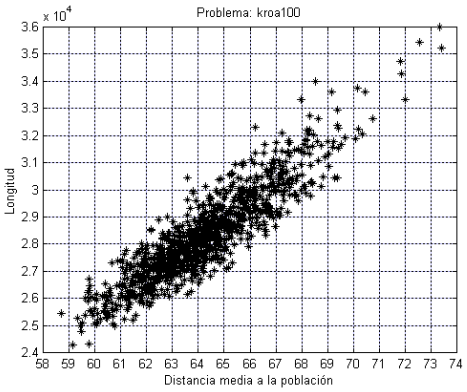


Figura 3.2: *l(r)* en función de *δ(P, r*)*

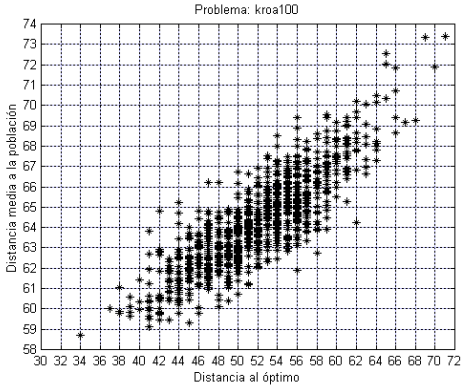


Figura 3.3: *δ(P, r*)* en función de *δ(r, r*)*

Figura 3: Problema *kroa100* usando *3-opt*

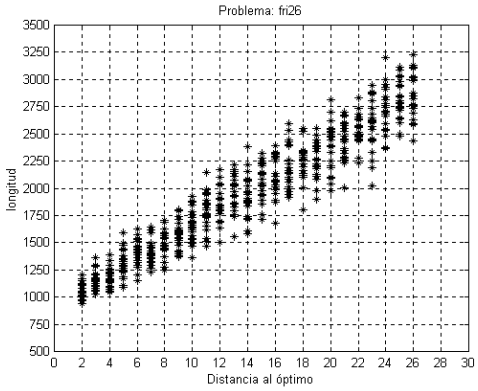


Figura 4.1: *l(r)* en función de *δ(r, r*)*

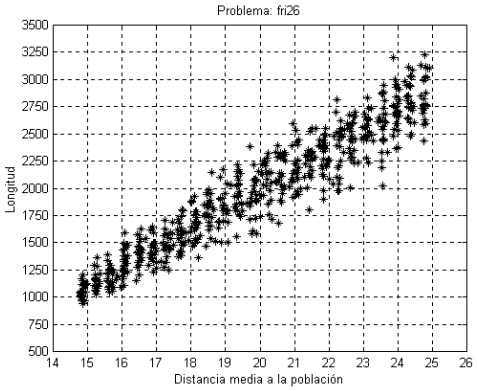


Figura 4.2: *l(r)* en función de *δ(P, r*)*

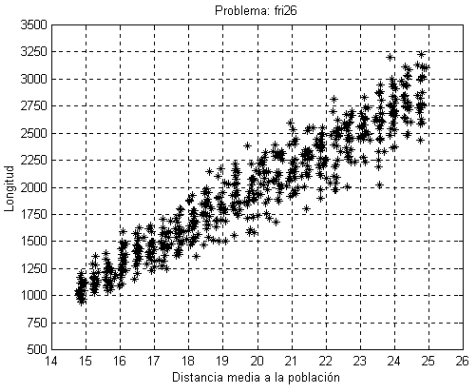


Figura 4.3 *δ(P, r*)* en función de *δ(r, r*)*

Figura 4: Problema *fri26* usando *Dist*

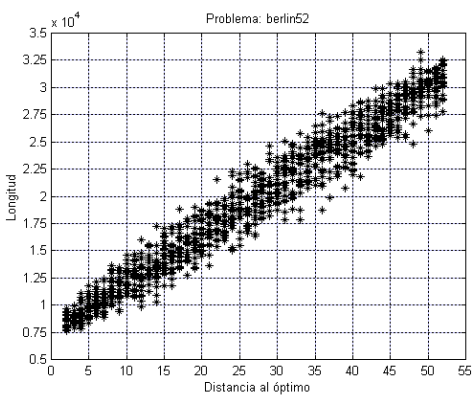


Figura 5.1: *l(r)* en función de *δ(r, r*)*

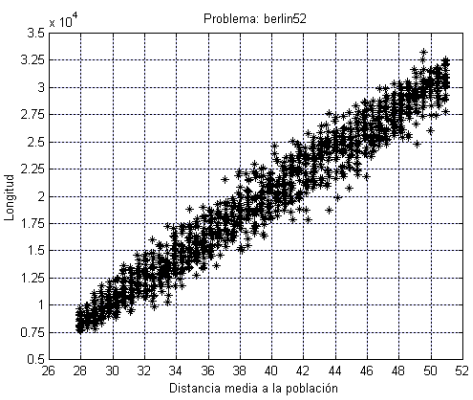


Figura 5.2 *l(r)* en función de *δ(P, r*)*

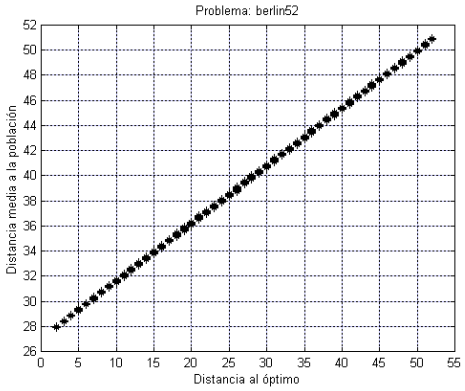


Figura 5.3: *δ(P, r*)* en función de *δ(r, r*)*

Figura 5: Problema *berlin52* usando *Dist*

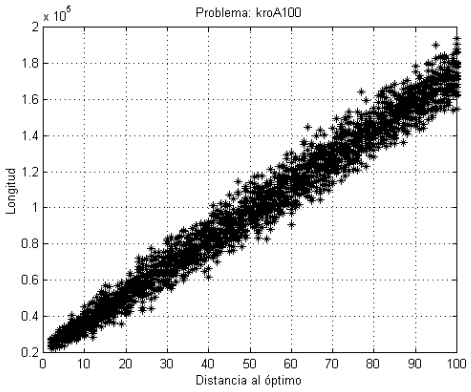


Figura 6.1: *l(r)* en función de *δ(r, r*)*

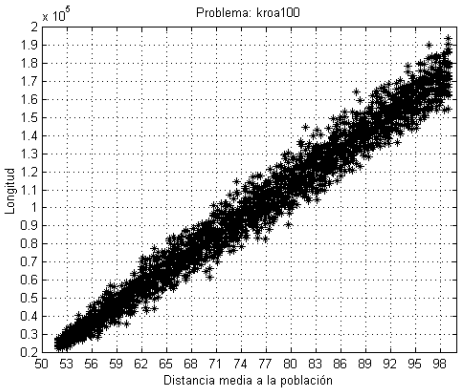


Figura 6.2: *l(r)* en función de *δ(P, r*)*

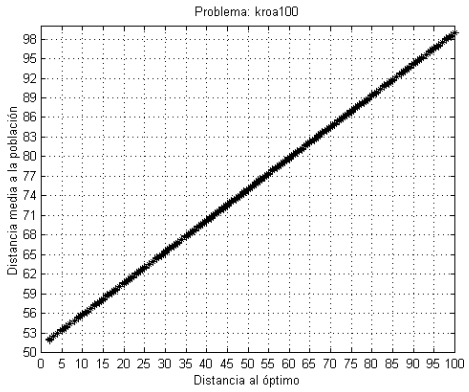


Figura 6.3: *δ(P, r*)* en función de *δ(r, r*)*

Figura 6: Problema *kroa100* usando *Dist*

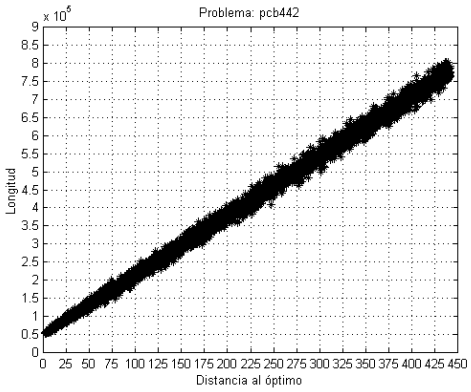


Figura 7.1: *l(r)* en función de *δ(r, r*)*

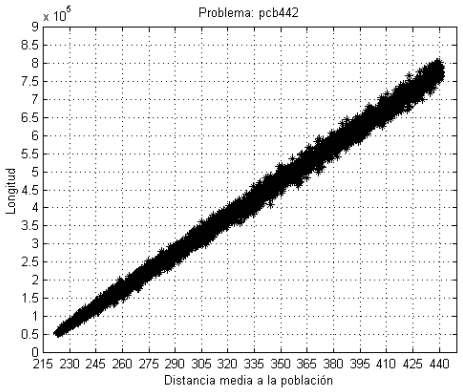


Figura 7.2: *l(r)* en función de *δ(P, r*)*

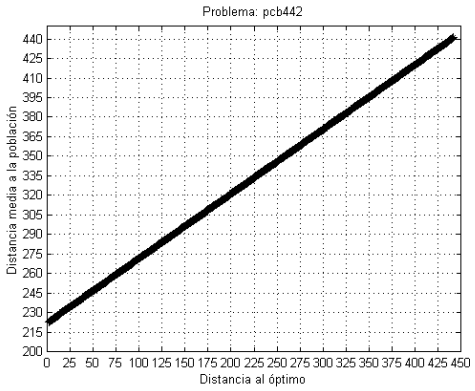


Figura 7.3: *δ(P, r*)* en función de *δ(r, r*)*

Figura 7: Problema *pcb442* usando *Dist*

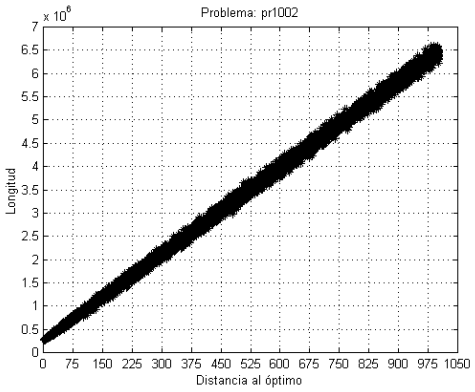


Figura 8.1: *l(r)* en función de *δ(r, r*)*

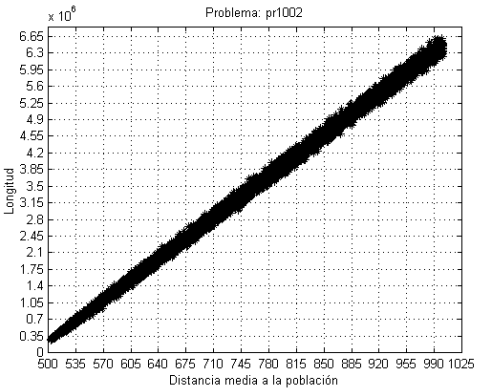


Figura 8.2: *l(r)* en función de *δ(P, r*)*

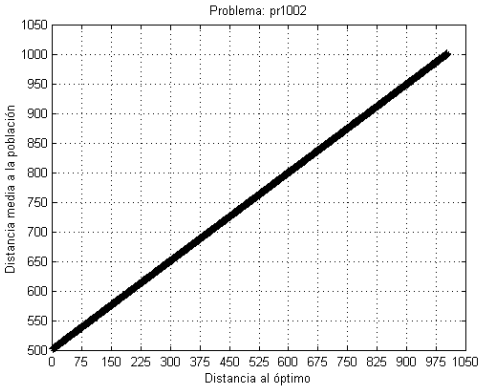


Figura 8.3: *δ(P, r*)* en función de *δ(r, r*)*

Figura 8: Problema *pr1002* usando *Dist*

5 Definición de espacio globalmente convexo

En una investigación previa, Boese [1] sugiere una interpretación intuitiva de un espacio globalmente convexo, haciendo una analogía con una estructura de *gran valle*, con un solo mínimo perceptible desde lejos, pero con picos y valles (mínimos locales) visto desde cerca. Una representación gráfica simplificada de esta interpretación puede observarse en la figura 9.

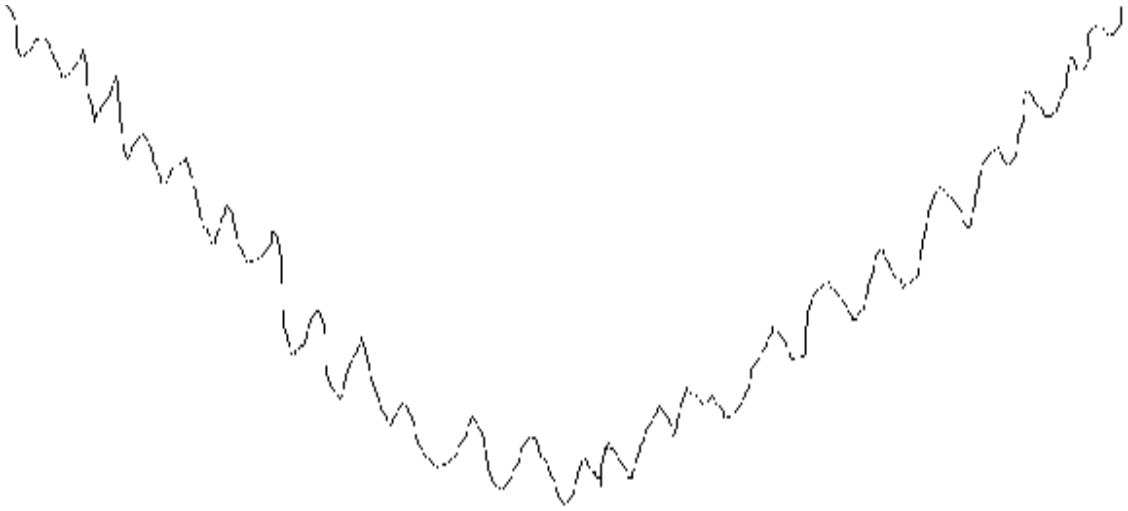


Figura 9: Representación gráfica de un espacio globalmente convexo.

Como puede notarse en la Figura 9, si tomamos una población uniformemente distribuida en el dominio de dicha función globalmente convexa, o entre los óptimos locales de esta, esta población tendría exactamente las mismas características que las encontradas en todos los problemas del TSP estudiados, es decir, se obtendrían valores muy significativos de las 3 correlaciones estudiadas y ampliamente graficadas en el presente trabajo:

1. $\rho(\delta(r,r^*);\ l(r))$ pues claramente las longitudes decrecen al acercarse a la solución óptima r^*.
2. $\rho(\delta(P,r);\ l(r))$ pues las mejores soluciones están en el centro del valle y las peores soluciones están en la periferia, conforme fuera explicado al final de la sección 2.
3. $\rho(\delta(r,r^*);\ \delta(P,r))$ como lógica consecuencia de las altas correlaciones anteriores, indicando además que el óptimo r^* estaría en una región central del espacio de búsqueda, rodeado de muy buenas soluciones.

En consecuencia, el presente trabajo presenta suficientes datos experimentales como para confirmar que la estructura del espacio de soluciones del *TSP* tiene en general una estructura globalmente convexa, conforme a lo sugerido sin demostración por Boese [1,2]. Lógicamente, no se pude afirmar que está estructura globalmente convexa se da necesariamente en todos los casos, pero definitivamente, los resultados experimentales demuestran que si se dan en la mayoría de los casos, y de hecho, los autores no lograron encontrar siquiera un problema del TSP que no sea globalmente convexo.

6 Conclusiones

Los resultados muestran unas muy altas correlaciones entre la $l(r)$ y su $\delta(r,r^*)$ en todas las poblaciones halladas con *Dist*. Esto claramente indica que a medida que un *tour* comparte un mayor número de arcos con el óptimo su longitud tiende a disminuir. Dicho de otra forma la posibilidad de hallar una buena solución es mayor mientras más cerca del óptimo r^* se busque. El estudio de la correlación entre $\delta(P,r)$ y $l(r)$ refuerza la idea de que las soluciones buenas se encuentran cerca unas de otras y en una zona central del espacio solución.
Esta propiedad resulta de especial importancia al intentar explicar el excelente desempeño de los algoritmos evolutivos basados en poblaciones, como las colonias de hormigas y los algoritmos genéticos, que al trabajar con una población que va optimizando el valor de $l(r)$, de acuerdo a lo aquí expuesto estaría concentrando su búsqueda en la zona central del espacio de búsqueda, donde en efecto se encuentran las mejores soluciones, y r^* en particular, dada la característica globalmente convexa del espacio de soluciones [3].
Cabe resaltar que las poblaciones halladas con la heurística de optimización local *3-opt* presentaron menores correlaciones que las halladas con *Dist*. Esto podría deberse a que son óptimos locales y no se hallan uniformemente distribuidas en el espacio de soluciones. Esto indica que la heurística al comenzar en puntos aleatorios fue

dirigiéndose automáticamente a la zona más cercana al óptimo. Es decir, que halló mejores soluciones a medida que encontraba arcos que coincidían con los del óptimo.
En resumen, el presente trabajo aporta por primera vez, amplios resultados experimentales que soportan la conjetura de un espacio de búsqueda globalmente convexo, en el caso del TSP, lo que explicaría el excelente desempeño de diversos algoritmos basados en poblaciones, a la hora de resolver el problema del cajero viajante.

Referencias

[1] Boese, K.D.: Cost versus Distance in the Traveling Salesman Problem. *Technical Report 950018*, University of California, Los Angeles, Computer Science Department, (1995).

[2] Boese, K.D, Kahng, A.B. y Muddu, S.: A new Adaptive Multi-Start Technique for Combinatorial Global Optimization, *Operations Research Center*. Vol 16, (1994). pp. 101-113

[3] Gómez O. y Barán B.: Arguments for ACO's Success, a ser publicado en "Genetic and Evolutionary Computation Conference – GECCO 2004". Seattle, Washington. Estados Unidos.

[4] Johnson D.S. y McGeoch L.A.: The traveling salesman problem: A case of study in local optimization, *Local Search in Comb. Optimization*, Eds. New York: Wiley: New York, (1997).

[5] Stützle, T. y Hoos, H.H.: Max-Min Ant System. *Future Generation Computer Systems.* Vol 16, (2000), pp. 889-914

[6] TSPLIB: http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/

Un compensador de Distorsión para Comunicaciones Inalámbricas

Nibaldo Rodríguez Agurto
Pontificia Universidad Católica de Valparaíso
Escuela de Ingeniería Informática
Av. Brasil 2241, Valparaíso-Chile
nibaldo.rodríguez@ucv.cl

Ricardo soto
Pontificia Universidad Católica de Valparaíso
Escuela de Ingeniería Informática
Av. Brasil 2241, Valparaíso-Chile
Ricardo.soto@ucv.cl
Wenceslao Palma
Pontificia Universidad Católica de Valparaíso
Escuela de Ingeniería Informática
Av. Brasil 2241, Valparaíso-Chile
Wenceslao.palma@ucv.cl

Abstract

Multilevel Qadrature Aplitude modulation (M-QAM) is of considerable interest for mobile communications, due to its high spectral efficiency. However, the major drawback of the technique is its bit error rate performance degradation in the presence of nonlinear amplification and Multipath fading (Rician/ Ryleight). In this paper, we propone a predistorter (PD) combined with iterative decoding in order to improve the bit error rate performance in the presence of both nonlinear and Rician fading effects. The PD is done at the transmitter side by a neural network within an extended Kalman filter (EKF) algorithm to estimate the coefficients and iterative decoding is done at the receiver side. The performance of the proposed scheme has been evaluated by computer simulation. The results show that the proposed method improves significantly the bit error rate.

Resumen:

La modulación Amplitud Cuadratura Multinivel (M-QAM) es de considerable interés para las comunicaciones móviles, debido a su alta eficiencia espectral. Sin embargo, la mayor desventaja de la técnica es su degradación de rendimiento de la razón de error de bit ante la presencia de amplificación no lineal y Multipath fading (Rician/Rayleigh). En este, artículo, proponemos un predistorsionador combinado con decodificación iterativa para mejorar el rendimiento de la razón de error de bit ante la presencia de amplificación no lineal y Rician fading. El PD es realizado en el lado del transmisor usando una red neuronal con un filtro de Kalman extendido para estimar los coeficientes de PD y el decodificador iterativo se realiza en el lado del receptor.

Keywords: predistortion, iterative decoding, neural network.

1.- Introducción

La modulación Amplitud Cuadratura Multinivel (M=4, 16 y 64-QAM) ha sido adoptada en varios estándares de comunicaciones inalámbricos, tales como: IEEE 802.11a y difusión de video digital por satélite (DVB-S) [7], debido a su alta eficiencia espectral. Sin embargo, este esquema de modulación es muy sensible y vulnerable a la distorsión no lineal de amplitud y fase introducida por el amplificador de potencia (AP) y el canal Multipath fading (Rician o Rayleigh) [12]. Como resultado de estas distorsiones, el rendimiento del receptor en términos de la razón de error de bit (BER) será degradado significativamente y el espectro de la señal transmitida será extendido a los canales adyacentes, provocando el efecto conocido como Interferencia de canales adyacentes (ACI). Una solución simple para evitar estos efectos negativos es operar el PA muy lejos del punto de operación óptimo o punto de saturación y así mantener los requerimientos de linealidad del sistema, pero esta solución ofrece un sistema de comunicación con baja eficiencia de potencia, logrando con ello que las baterías de los equipos móviles tengo una baja duración. Sin embargo, hoy en día los usuarios demandan equipos móviles de bajo consumo de potencia y amplio ancho de banda.

Por lo tanto, para satisfacer la demanda de los usuarios han sido propuestas dos técnicas de compensación durante las últimas décadas, las cuales son conocidas con el nombre de Predistorsión (PD) y Ecualización. La PD se implementa en el transmisor usando PD de la data o PD de la señal y la ecualización se implementa en el receptor usando un esquema de ecualización no lineal.

En este artículo proponemos un esquema de PD combinado con Decodificación Iterativa para compensar los efectos de distorsión no lineal, distorsión multiplicativa Rician fading y también para reducir el ruido Gaussian blanco aditivo (RGBA). El PD es usado para compensar la distorsión no lineal introducida por el PA y está basado en una red neuronal con una capa oculta y un nodo para la entrada y salida. Los coeficientes del PD son ajustados usando el filtro de Klaman Extendido [14]. El esquema de Decodificación Iterativa está basado en el uso de un Turbo Código (TC). Los TC fueron presentados por primera vez en la Conferencia Internacional de Comunicaciones celebrada en Génova, por los investigadores Berrou, Glavieux y Thitimajshima [4], en esa oportunidad los TC sólo fueron presentados para canales Gaussian. Los TC son una familia de códigos concatenados en paralelo. Los códigos constituyentes de los TC son códigos convolucionales y según [4], los códigos constituyentes que proporcionan mejores prestaciones son los Códigos Convolucionales Recursivos Sistemáticos (CRS). El proceso de decodificador se compone de tantos decodificadores constituyentes como códigos constituyentes tenga el TC y cada uno de ellos calcula una distribución de probabilidad a posteriori (DPAP) a partir de los símbolos demodulados. El algoritmo central de cada bloque se baso en el algoritmo BCJR propuesto en la referencia [1]. Este algoritmo, no es implementable porque requiere de toda la secuencia de símbolos demodulados para poder proporcionar la secuencia de DPAP. Para evitar este problema en las referencias [2][3] y [8] ha sido propuesto un algoritmo BCJR modificado, el cual ha sido ampliamente usado durante los últimos años para el proceso de turbo decodificación.

Un aspecto importante en el diseño de un TC es el esquema de intercalación de bits, puesto que el intercalador determina el comportamiento asintótico del TC. Esta característica ha sido motivo para que algunos investigadores durante la última década hayan investigado varios tipos de intercaladores de bits para mejorar el comportamiento asintótico que ofrece Turbo Código. Entre estos investigadores se destacan los trabajos de Divsalar [6]. Divsalar, propuso un tipo de intercalador de bits denominado *S*-Random, donde el parámetro S se selecciona como $S \leq \sqrt{L/2}$, donde L representa el tamaño del intercalador de bits. Además, cuando el parámetro L crece, el rendimiento del TC mejora significativamente, pero la complejidad del Turbo Decodificador también crece [3]. Para mantener un buen equilibrio entre complejidad y la razón de error de bits, un tamaño recomendable es seleccionar $L \leq 1024$.

Finalmente, este artículo está organizado de la siguiente forma: en la sección 2 se describe la funcionalidad de cada uno de los componentes del transmisor y receptor del sistema de comunicación. La sección 3, se describe la estructura y el algoritmo de Predistorsión. En la sección 4, se presenta la evaluación de rendimiento lograda a través de simulación computacional usando modulación QPSK (4-QAM) y por último, las principales conclusiones son dadas en la sección 5.

2.- Descripción del sistema propuesto

En la Figura 1, se presenta el diagrama de bloque del sistema de transmisión propuesto para señales banda base QPSK. La entrada al sistema es una secuencia de bits de información denotada por la variable $d_k \in \{0,1\}_{k=1}^{N}$. Los d_k son codificados usando el esquema de codificación denominado Turbo Códigos. El TC usado en este artículo está formado por dos códigos convolucionales recursivos sistemáticos unidos en forma paralela y separado por un intercalador de bits. Durante el proceso de turbo codificación la secuencia de bits de información d_k se codifica usando el primer codificador, el cual genera una secuencia de salida formada por bits sistemáticos $d_k = x_k^s$ y bits de paridad $x_{1,k}^p$. Posteriormente los bis d_k son intercalado usando un esquema de intercalación de bits y la salida del intercalar $\pi(d_k) = \widetilde{d}_k$ se utiliza como entrada en el segundo codificador CRS, nuevamente este codificador genera una salida formada por bits sistemáticos y bits de paridad $x_{2,k}^p$. Para lograr un esquema de codificación con razón igual a un medio se usa un dispositivo denominado selector (no mostrado en la Figura 1), el cual selecciona los bits de paridad impar del primer codificador y los bits de paridad par del segundo codificador $x_k^p = \left\{x_{1,1}^p, x_{2,2}^p, x_{1,3}^p, x_{1,4}^p, x_{1,5}^p, x_{1,6}^p, \ldots\right\}$. Luego, la secuencia de salida $\left\{x_k^s, x_k^p\right\}$del turbo codificador se transforma en un símbolo complejo M-QAM usando el esquema de Gray. Estos símbolos son filtrados usando un filtro raíz cuadrada *raised cosine*, el cual se implementa usando 25 coeficientes, un factor *roll-off* igual a 0.5. El lector puede encontrar una descripción más detallada sobre el filtro transmisor en la referencia [11]. Posteriormente, la señal banda base $x(t)$ es predistorsionada por el dispositvo de Predistorsión y luego amplificada por un AP del tipo TWT [13]. La señal amplificada es representada por:

$$z(t) = A(y_\rho)\exp\left(j\cdot(y_\theta + P(y_\rho))\right) \tag{1}$$

donde y_ρ e y_θ representan la amplitud y fase de la señal predistorsionada $y(t)$. La funciones $A(.)$ y $P(.)$ denotan las conversiones de amplitud y fase no lineal; respectivamente.

Para un amplificador TWT, las funciones de amplitud fase son dadas por [13]:

$$G(y_\rho) = \frac{2\cdot y_\rho}{1+y_\rho^2} \tag{2}$$

$$\Phi(y_\rho) = \frac{2\cdot y_\rho^2}{1+y_\rho^2}\cdot\frac{\pi}{6} \tag{3}$$

La distorsión no lineal de AP depende de la potencia *Backoff* (BO). La potencia OB de entrada es definida como la razón entre la potencia de saturación de entrada $P_{sat,i}$ y la potencia promedio de entrada $P_{pr,i}$:

$$IBO = 10\log_{10}\left(\frac{P_{sat,i}}{P_{pr,i}}\right) \tag{4}$$

Finalmente, en el transmisor la señal amplificada $z(t)$ se propaga sobre un canal móvil, la cual también se deteriora por la presencia de ruido Gaussian blanco aditivo (RGBA). En este artículo, consideramos un canal *Rician flat fading* $c(\tau,t)$. Este modelo se justifica por la existencia de una ruta directa entre el transmisor y/o el receptor en la mayoría de los casos. La respuesta de impulso de tiempo variable del canal $c(\tau,t)$ se modela como un proceso Gaussian complejo con un valor medio distinto de cero y el envolvente del $c(\tau,t)$ en un instante de tiempo t posee una distribución de *Rice*. Desde el modelo de Jake [10], la función de autocorrelación de $c(\tau,t)$ es dada como:

$$R_c(\tau) = \frac{1}{2(K_R+1)} J_0(2\pi f_D \tau) \tag{5}$$

donde $J_0(.)$ es la función de Bessel de orden cero de primer tipo y f_D representa la frecuencia Doppler máxima entre el transmisor y el receptor, la cual está relacionada con la velocidad del vehículo v y la frecuencia *carrier* f_c. La valor $f_n = f_D \tau$ es comúnmente conocido como frecuencia Doppler normalizada y el parámetro K_R es interpretado generalmente como la razón de potencia entre la componente directa y la componente difusa.

En un instante de tiempo t la señal recibida se representa como:

$$r(t) = c(t)z(t) + n(t) \tag{6}$$

donde $c(t)$ representa la ganancia del canal y $n(t)$ corresponde al RGBA complejo con densidad espectral igual a $N_o/2$.

La señal recibida $r(t)$ es pasada a través del filtro receptor y luego es muestreada. La secuencia de salida muestreada p_k se constituye en la entrada del proceso denominado *Demapping*. El *Demapping* divide los símbolos complejos p_k en dos secuencias $\{y_k^s\}$ y $\{y_k^p\}$ y luego estas secuencias son demoduladas independientemente contra sus respectivas cotas de decisiones. En el proceso de turbo decodificación existe un dispositivo llamado de-selector (no mostrado en la Figura 1), el cual realiza el proceso inverso del selector. Este dispositivo convierte los bits de paridad recibido y_k^p en una secuencia paralela $\{y_{1,k}^p, y_{2,k}^p\}$ y agrega un valor cero en los bis de paridad no transmitidos. La secuencia de bits paralelo recibida $\{y_k^s, y_{1,k}^p, y_{2,k}^p\}$ se decodifica usando el proceso turbo decodificador, el cual utiliza un algoritmo sub-óptimo conocido como Maximum a Posteriori (MAP). La derivación de este algoritmo ha sido bien documentada en la referencia [2][3] y [8].

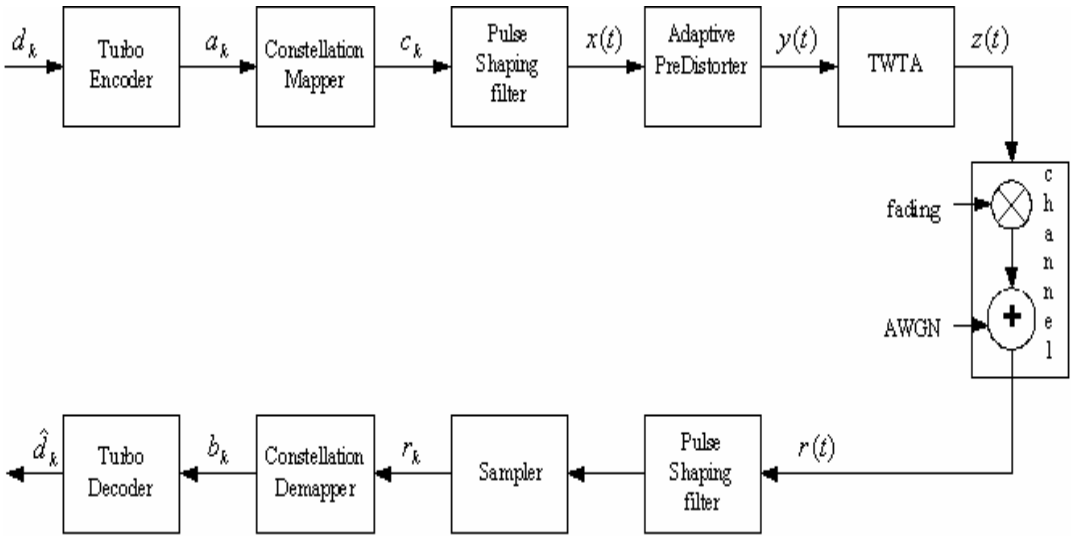


Figura 1 Sistema de transmisión banda base.

3.- Estructura y Algoritmo de Predistorsión

Consideremos una representación banda base para las señales de entrada al PD $x(t)$, señal de salida del PD $y(t)$ y señal de salida del AP $z(t)$:

$$x(t) = x_\rho \exp(j \cdot x_\theta) \tag{7}$$

$$y(t) = y_\rho \exp(j \cdot y_\theta) \tag{8}$$

$$z(t) = z_\rho \exp(j \cdot z_\theta) \tag{9}$$

donde x_θ, y_θ y z_θ representan la fase de las señales y x_ρ, y_ρ y z_ρ denotan la amplitud de las señales de las ecuaciones anteriores.

La salida del PD será representada en amplitud y cuadratura como:

$$y(t) = \mathrm{M}(x_\rho(t)) \exp\left[j(x_\theta(t) + \mathrm{N}(x_\rho))\right] \tag{10}$$

Usando la ecuación (1) y (6) se obtiene la señal de salida del AP como:

$$z(t) = \mathrm{A}\left[\mathrm{M}(x_\rho(t))\right] \exp\left[j(x_\theta + \mathrm{N}(x_\rho) + P(\mathrm{M}(x_\rho(t))))\right] \tag{11}$$

Para obtener un PD ideal, la señal $z(t)$ debe ser igual o proporcional a la señal $x(t)$. Esto es:

$$\begin{aligned} &\mathrm{A}\left[\mathrm{M}(x_\rho(t))\right] = \alpha \cdot x_\rho(t) \\ &\mathrm{N}\left[x_\rho(t)\right] = -P\left[\mathrm{M}(x_\rho(t))\right] \\ &\alpha^2 = \beta \cdot P_{sat}, \quad 0 < \beta \le 1 \end{aligned} \tag{12}$$

donde P_{sat} representa la potencia de saturación. En este artículo la potencia de saturación es $P_{sat} = 1$.

Por lo tanto, la máxima potencia de salida del amplificador linealizado será igual a $\beta \cdot P_{sat}$ [5].

Finalmente, la señal de salida del PD será obtenido por:

$$y(t) = \mathrm{A}^{-1}(x_\rho(t)) \exp\left[j\left(x_\theta(t) - P(\mathrm{A}^{-1}(x_\rho(t)))\right)\right] \tag{13}$$

Por lo tanto, para obtener la función de Predistorsión $y(t) = f_{pd}(.)$, es sólo necesario encontrar la función inversa de $\mathrm{A}(.)$. Para hallar la función inversa es usada una rede neuronal, puesto que las redes neuronales cumple la propiedad de aproximadotes universales [9].

Para implementar el proceso de aprendizaje de PD se requiere formar una base de datos compuesta por muestras de la función de amplitud $\mathrm{A}\left[x_\rho(n)\right]$. Esto es: $\mathrm{B} = \{z_\rho(n), x_\rho(n); n = 1,2,..., N_s\}$, donde z_ρ representa la entrada a la red neuronal, x_ρ representa la salida deseada y N_s denota el número de muestras de la función de amplitud del AP.

El problema de encontrar los pesos óptimos de PD pueden ser tratado como un problema de identificación de parámetros. Asumamos que todos los pesos de PD son son representados en un vector $w = \left[a_1, a_2, ..., a_{N_h}, b1, ..., b_{N_h}\right]$. Entonces usando la siguiente ecuación podemos hallar los pesos de PD:

$$\begin{aligned} w(n+1) &= w(n) \\ x_\rho(n) &= f_{PD}\left[z_\rho(n), w(n)\right] + v(n) \end{aligned} \tag{14}$$

donde es asumido que el estado inicial $w(0)$ y la secuencia $\{v(n)\}$ son idenpendientes.

Las ecuaciones del EKF para ajustar los coeficientes de PD son [14]:

$$K(n) = P(n-1)F(n)\left[R(n) + F^T(n)P(n-1)F(n)\right]^{-1} \tag{15}$$

$$P(n) = P(n-1) - K(n)F^T(n)P(n-1) \tag{16}$$

$$\hat{w}(n+1) = \hat{w}(n) + K(n)\left[x_\rho(n) - f_{PD}(z_\rho(n), \hat{w}(n))\right] \tag{17}$$

donde $F(n)$ representa la matriz Jocobiana definida por:

$$F(n) = \frac{\partial f_{PD}}{\partial w}\Big|_{w=\hat{w}(n-1)} = \left(\frac{\partial f_{PD}}{\partial w_j}\right), \; j = 1,..., N_{h,} N_h + 1,...,2N_h \tag{18}$$

y $K(n)$ representa la matriz de ganancia de Kalman.

4.- Discusión de Resultados

En esta sección, el rendimiento del esquema propuesto fue evaluado a través de simulación computacional usando una secuencia de bits de información de largo $N_{bit} = 100000$. El tipo de modulación seleccionada fue QPSK, el intercalación de bits usado fue el propuesto en las referencias [6] y el tamaño de intercalación de bits fue igual a 512 y el parámetro *S=13*. La decisión o estimación de los bits fue tomada en la salida del proceso turbo decodificación, donde los bits decodificados $\{\hat{d}_k\}$ fueron comparados con los bits transmitidos $\{d_k\}$ para calcular la razón de error de bits versus la razón señal a ruido. Cabe hacer nota que estos parámetros han sido mantenidos constantes en cada una de las simulaciones presentadas en este artículo.

Los coeficientes de PD neuronal fueron estimados durante el proceso de aprendizaje usando $N_s = 100$ muestras de la función de amplitud del AP y el punto de operación del AP fue igual a *IBO*= -1 dB. La estructura de PD neuronal fue configurada con un nodo de entrada, uno de salida y cinco nodos para la capa oculta. Esta estructura es denota por PDTC(1,5,1). La inicialización de los coeficientes de PD fue usando un proceso random gaussiano con distribución normal $\mathrm{N}(0,1)$. El número de iteraciones o épocas del aprendizaje fue igual a 50 y el error cuadrático medio fue igual a 3×10^{-4}.

El canal móvil fue modelado como un proceso Gaussian complejo con un valor medio distinto de cero y es representado por la siguiente ecuación:

$$c(n) = \left|u(n) + v(n)\right| \tag{19}$$

donde $u(n)$ representa la componente difusa y $v(n)$ representa la componente directa.

Para generar la secuencia difusa $u(n)$ ha sido utilizado el esquema propuesto por David Young en [15], el cual permite generar variables aleatorias correlacionadas usando la transformada de Fourier discreta inversa. La componente difusa es dada por la siguiente expresión:

$$u(n) = \frac{1}{N}\sum_{k=0}^{N-1}\left(F(k)A(k) - jF(k)B(k)\right)e^{j(2\pi kn/N)} \tag{20}$$

donde cada elemento de la sucesión $\{A(k)\}$ y $\{B(k)\}$; $k = 0,1,...,N-1$, tiene una distribución normal con media igual a cero y varianza σ^2. La secuencia $\{F(k)\}$ representa los coeficientes de un filtro de repuesta de impulso finito dado por la ecuación (21).

$$F(k) = \begin{cases} 0, & k = 0 \\ \sqrt{\dfrac{1}{2\sqrt{1-\left(\dfrac{k}{Nf_n}\right)^2}}}, & k = 1,2,...,k_n - 1 \\ \sqrt{\dfrac{k_n}{2}\left[\dfrac{\pi}{2} - \arctan\left(\dfrac{k_n - 1}{\sqrt{2k_n - 1}}\right)\right]}, & k = k_n \\ 0, & k = k_n + 1,...,N - k_n - 1 \\ \sqrt{\dfrac{k_n}{2}\left[\dfrac{\pi}{2} - \arctan\left(\dfrac{k_n - 1}{\sqrt{2k_n - 1}}\right)\right]}, & k = N - k_n \\ \sqrt{\dfrac{1}{2\sqrt{1-\left(\dfrac{N-k}{Nf_n}\right)^2}}}, & k = N - k_n + 1,...,N-1 \end{cases} \quad (21)$$

donde $k_n = \lfloor Nf_n \rfloor$ y $\lfloor . \rfloor$ representa la función parte entera.

La componente directa $v(n)$ se define por:

$$v(n) = \sqrt{2K_R \operatorname{var}[\operatorname{Re}(u(n))]} \quad (22)$$

donde las funciones $\operatorname{var}(.)$ y $\operatorname{Re}(.)$ representas la varianza y la parte real de un número complejo; respectivamente.

Para evaluar el rendimiento del canal móvil con Rician fading y ruido gaussino, asumimos que las muestras son simuladas usando la ecuación (19). Los parámetros del canal son: frecuencia portadora 2.5Ghz, frecuencia de muestreo 100kHz, factor rician K_R=4dB. Las velocidades del móvil fueros 32.4 km/h y 36.72km/h. estas velocidades corresponde a una frecuencia Doppler normalizada igual a $f_n = 0.00075$ y $f_n = 0.00085$; respectivamente.

En la Figura 2 se presenta el rendimiento de la razón de error de bits (BER) versus la razón señal a ruido (SNR) considerando tres casos. a) sin técnica de compensación, b) sólo con decodificación iterativa (turbo código) y c) con Predistorsión y turbo código PDTC(1,5,1). Desde esta figura se puede apreciar que los rendimientos logrados con los casos a) y b) ofrecen un BER muy pobre para una SNR entre 1.5dB y 5.5dB. Este resultado permite concluir que los Turbo códigos no son eficientes para corregir errores de bits provocados por un canal No Lineal Rician Fading.

El rendimiento logrado con el caso c) es significativamente mejor cuando se compara con los casos a) y b). Esta ganancia de SNR se logra como consecuencia de la incorporación de un compensador de distorsión no lineal.

Desde la Figura 2 se pude observar que cuando la frecuencia Doppler crece desde $f_n = 0.00075$ a $f_n = 0.00085$ el rendimiento del BER versus SNR mejor considerablemente y la ganancia lograda es de aproximadamente 1.8 dB de SNR para un BER=10^{-4}. Por lo tanto, estos resultados ilustran que el compensador propuesto PDTC(1,5,1) permite lograr un rendimiento del BER cuando el proceso fading es menos correlacionado.

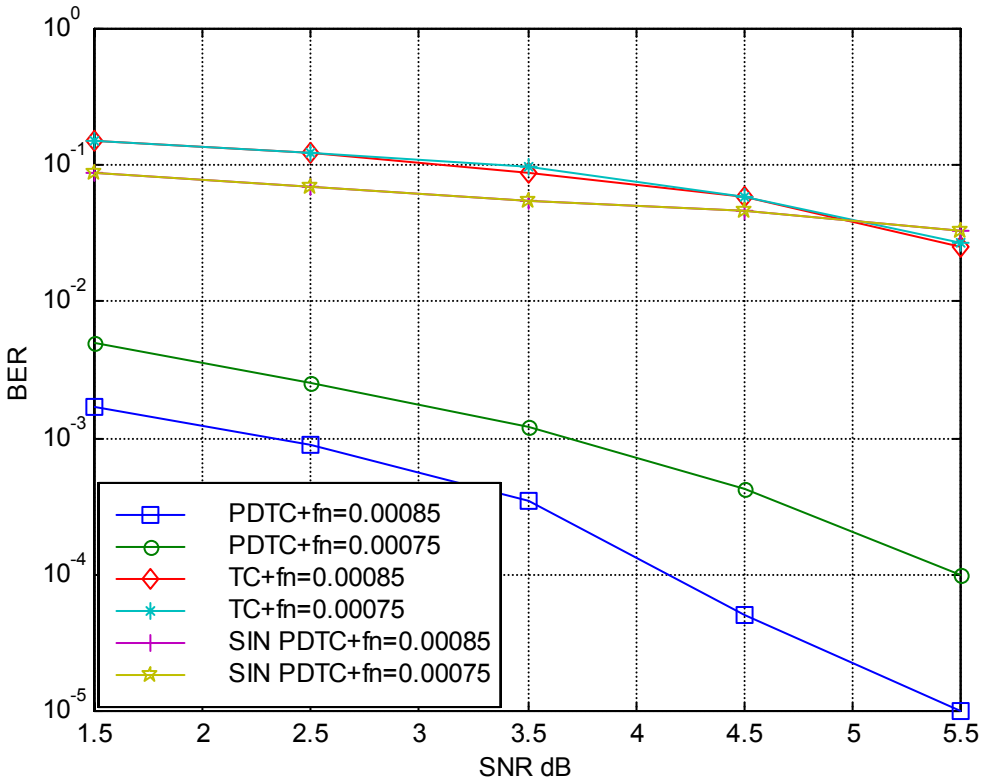


Figura 2 BER vs SNR para QPSK con *IBO*= -1dB, Rician fading K_R=4dB y Ruido gaussiano.

4.- Conclusiones

En este artículo un esquema de compensación de distosión no lineal y Rician fading fue presentado. La estructura del compensador propuesto está formado por un predistorsionador y un turbo código.

Los resultados obtenidos a través de simulación computacional permiten concluir que el compensador propuesto logra una ganancia significativa cuando se compara con un sistema de comunicación sin técnicas de compensación. Para el caso de señales QPSK el compensador ofrece una ganancia de 1.8 dB de SNR para un BER=10^{-4}.

Agradecimientos

El autor agradece la ayuda económica ofrecida por el proyecto DGI. Nº 209.730/2004, de la Pontificia Universidad Católica de Valparaíso, Chile.

Referencias

[1] Bahl L.R., Coke J., Jelinek F., Raviv J., *Optimal decoding of linear codes for minimising symbol error rate*, IEEE Trans. Information Theory, Vol. IT-20, pp. 284-287, 1974.

[2] Benedetto S., D. Divsalar, G. Montorsi and F. Pollara MAP algorithms and their applications to decode parallel and serial code concatenations, *in Fifth ESA Int. Workshop on Digital Signal Proc. Tech. Applied to Space Communications*, pp. 8.10-8.24, 1996.

[3] Benedetto S. and Montersi G., *Design of parallel concatenated convolutional codes*, IEEE Trans. Commun., Vol. 44, Nº 5, pp. 591-600,May 1996.

[4] Berrou C., Glavieux A., and Thitimajshima P., *Near Shannon limit error-correcting coding and decoding: Turbo Codes*, in Proc. ICC'93, pp.1064-1070, May 1993.

[5] Cavers J.K., *Amplifier linearization using a digital predistorter with fast adaptation and low memory requirements*, IEEE Trans. Veh. Tech., Vol. 39, Nº 4, pp. 374-382, Nov. 1990.

[6] Divsalar D. and Pollara F., *Turbo codes for PCS applications*, IEEE Int. Conf. Commun., pp. 54–59, 1995.

[7] ETSI, *Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for 11/12 GHz Satellite Services*, EN 300 421 v.1.1.2, August 1997.

[8] Hagenauer J., Offer E., and Papke L., *Iterative decoding of binary block and convolutional codes*, IEEE Trans. Information Theory, Vol. 42, pp.429-445, March 1996.

[9] Kornik K., Stichcombe M. and White H., *Multilayer feedforward networks are universal approximators*, Neural Networks, Vol. 2, pp. 359-366, 1989

[10] Jakes W. C., *Microwave mobile communications,* New York, Wiley, 1974.

[11] Proakis John, *Digital communications,* third edition, McGraw-Hill, 1995.

[12] Rodriguez N., Soto I., and Carrasco R., *.Adaptive predistortion of COFDM signals for a mobile satellite channel,* International Journal of Communication systems, Vol.16, pp. 137-150, March 2003.

[13] Saleh M. y Adel A, *Frecuency-Independent and Frecuency-Dependent nolinear models TWT amplifiers*, IEEE Trans. Comm., Vol. COM-29, pp. 1715-1719, November 1981.

[14] Singhal S. and Wu L., *Training feedforward networks with the extended Kalman algorithm*, in ICASSP-89, 1989 Int. Conf. Acoust., Speech, Signal Processing, Vol. 2, pp. 1187–1190, 1989

[15] Young D. and Beaulieu C., *The generation of correlated Rayleigh random variates by inverse discrete Fourier transform*, IEEE Trans. Commun., Vol. 48, Nº 7, pp. 114-1127, July 2000.

Relationship between Genetic Algorithms and Ant Colony Optimization Algorithms

Osvaldo Gómez
Universidad Nacional de Asunción
Centro Nacional de Computación
Asunción, Paraguay
ogomez@cnc.una.py

and

Benjamín Barán
Universidad Nacional de Asunción
Centro Nacional de Computación
Asunción, Paraguay
bbaran@cnc.una.py

Abstract

Genetic Algorithms (GAs) were introduced by Holland as a computational analogy of adaptive systems. GAs are search procedures based on the mechanics of natural selection and natural genetics. Ant Colony Optimization (ACO) is a metaheuristic inspired by the foraging behavior of ant colonies. ACO was introduced by Dorigo and has evolved significantly in the last few years. Both algorithms have shown their effectiveness in the resolution of hard combinatorial optimization problems. This paper shows the relationship between these two evolutionary algorithms. This relationship extends the reasons of ACO's success in TSP to GAs. Finally, the significance of the crossover and the genetic diversity in globally convex structures is explained.

Keywords: Artificial Intelligence, Ant Colony Optimization, Genetic Algorithms, Reasons for success.

1 Introduction

A Genetic Algorithm (GA) is a randomized search method modeled on evolution and introduced by Holland [8]. GAs are being applied to a variety of problems and becoming an important tool in combinatorial optimization problems [5]. Ant Colony Optimization (ACO) is a metaheuristic inspired by the foraging behavior of ant colonies [2]. In the last few years, ACO has empirically shown its effectiveness in the resolution of several different NP-hard combinatorial optimization problems [3]. So, GAs and ACO are evolutionary algorithms inspired by different nature phenomena.

One example of a well known NP-hard combinatorial optimization problem is the Traveling Salesman Problem (TSP). This paper uses the TSP as a case study problem as previous works did [2, 4, 11]. Previous studies also proposed a globally convex or big valley structure of the TSP solution space [1, 4].

A new version of ACO designed by the authors, Omicron ACO (OA), has shown to be an useful analysis tool to study the reasons of ACO's success [4]. OA also inspired the authors to find the relationship between ACO algorithms and GAs. This relationship allows the extension of the reasons of ACO's success to GAs in globally convex structures. Through this extension, the significance of the crossover and the genetic diversity in GAs can be understood.

The TSP is summarized in Section 2. In Section 3, the Simple Genetic Algorithm (SGA) and the Omicron Genetic Algorithm (OGA) are described. New strategies for OGA are proposed in Section 4. The standard ACO approach, the Population-based ACO (P-ACO) and the Omicron ACO are presented in Section 5. The relationship between GAs and ACO algorithms is shown in Section 6. The extension of the reasons of ACO's success to GAs is explained in Section 7. Finally, the conclusions and future work are given in Section 8.

2 Study Problem

In this paper the symmetric Traveling Salesman Problem (TSP) is used as a case study problem for comparing the analyzed algorithms. The TSP is a hard combinatorial optimization problem, easy to understand, which has been considerably studied by the scientific community. Researchers have applied GAs and ACO algorithms successfully to this problem [2, 5, 11].

The TSP can be represented by a complete graph $G = (N, A)$ with N being the set of nodes, also called cities, and A being the set of arcs fully connecting the nodes. Each arc (i, j) is assigned a value $d(i, j)$ which represents the distance between cities i and j. The TSP is the problem of finding the shortest closed tour visiting each of the $n = |N|$ nodes of G exactly once. For symmetric TSPs, the distances between the cities are independent of the direction of traversing the arcs, that is $d(i, j) = d(j, i)$ for every pair of nodes. Suppose that r_x and r_y are TSP tours or solutions over the same set of n cities. For this work, $l(r_x)$ denotes the length of tour r_x. The distance $\delta(r_x, r_y)$ between r_x and r_y is defined as n minus the number of edges contained in both r_x and r_y.

3 Genetic Algorithms

Genetic Algorithms (GAs) were introduced by Holland as a computational analogy of adaptive systems [8]. GAs are search procedures based on the mechanics of natural selection and natural genetics. Next, the Simple Genetic Algorithm (SGA) presented by Goldberg in [5] - which may be considered as a standard approach - is described.

3.1 Simple Genetic Algorithm

The Simple Genetic Algorithm (SGA) has a population P of p individuals P_x representing solutions of an optimization problem. The SGA uses a binary codification of these solutions or individuals represented by strings. In a maximization context, the value of the objective function of every individual of P is considered its fitness. An initial population is chosen randomly, then a set of simple operations that uses and generates successive Ps is executed iteratively. It is expected that P improves over time until an end condition is reached.

The operators of the SGA are reproduction, crossover and mutation. Reproduction is a process in which p individuals are selected with a probability proportional to their fitness to be parents. Crossover is a process in which 2 different parents are iteratively selected from the set of p parents to swap information between them to generate 2 new individuals (offspring). This is done choosing randomly a point of break for the parents and swapping parts between them. Then mutation is applied to every offspring. Mutation is the alteration of the bits of an individual with a small predefined probability, sometimes known as mutation coefficient (mc). These new altered individuals compose the new population P.

Next, the main pseudocode of the SGA is presented.

Pseudocode of the *Simple Genetic Algorithm*

```
g = 0                                  /* Initialization of the generation counter */
P = Initialize population ()           /* Random generation of p individuals */
REPEAT UNTIL end condition
  F = Reproduction (P)                 /* Selection of p parents through a roulette */
  S = Crossover (F)                    /* Swap of information between different pairs of parents */
  M = Mutation (S)                     /* Alteration of individuals' bits with probability mc */
  P = Update population (M)            /* M is copied to P (P = M) */
  g = g + 1                            /* Increment of the generation counter */
```

3.2 Omicron Genetic Algorithm

The literature in evolutionary computation has defined a great variety of GAs that maintain the same philosophy, varying operators and adding different principles like elitism [5, 10]. Using the Simple Genetic Algorithm as a reference, this Section presents a new version, the Omicron Genetic Algorithm (OGA), a Genetic Algorithm designed specifically for the TSP.

3.2.1 Codification

The OGA has a population P of p individuals or solutions, as the SGA does. Every individual P_x of P is a valid TSP tour and is determined by the arcs (i, j) that compose the tour. Unlike the SGA, that uses a binary codification, the OGA uses an n-ary codification. Considering a TSP with 5 cities $c1$, $c2$, $c3$, $c4$ and $c5$, the tour defined by the arcs $(c1, c4)$, $(c4, c3)$, $(c3, c2)$, $(c2, c5)$ and $(c5, c1)$ will be codified with a string containing the visited cities in order, i.e. $\{c1, c4, c3, c2, c5\}$.

3.2.2 Reproduction

The OGA selects randomly two parents (F_1 and F_2) from the population P, as does an SGA reproduction. The selection of a parent is done with a probability proportional to the fitness of each individual P_x, where $fitness(P_x) \propto 1/l(P_x)$. Unlike the SGA, where two parents generate two offspring, in the OGA, both parents generate only one offspring. In the SGA, p offspring are obtained first to completely replace the old generation. In the OGA, once an offspring is generated, it replaces the oldest element of P. Thus, the population will be a totally new one in p iterations and it would be possible to consider this population a new generation. In conclusion, the same population exchange as in the SGA is made in the OGA, but in a progressive way.

3.2.3 Crossover and Mutation

The objective of crossover in the SGA is that the offspring share information of both parents. In mutation, the goal is that new information is added to the offspring, and therefore is added to the population. In the SGA, the operators crossover and mutation are done separately. To facilitate the obtaining of offspring who represent valid tours, the crossover and the mutation operators are done in a single operation called Crossover-Mutation in OGA. Even so, the objectives of both operators previously mentioned will stay intact.

To perform Crossover-Mutation, the arcs of the problem are represented in a roulette, where every arc has a weight w or a probability to be chosen. Crossover-Mutation gives a weight w of 1 to each arc (i, j) belonging to set A, i.e. $w_{ij} = 1\ \forall (i, j) \in A$. Then, a weight of $O/2$ is added to each arc (i, j) of F_1, i.e. $w_{ij} = w_{ij} + O/2\ \forall (i, j) \in F_1$, where Omicron ($O$) is an input parameter of the OGA. Analogously, a weight of $O/2$ is added to each arc (i, j) of F_2. Iteratively, arcs are randomly taken using the roulette to generate a new offspring. While visiting city i, consider $\mathcal{N}_i$ as the set of cities not yet visited and that allows the generation of a valid tour. Therefore, only the arcs $(i, j)\ \forall j \in \mathcal{N}_i$ participate in the roulette, with their respective weights w_{ij}. Even so the crossover is done breaking the parents and interchanging parts in the SGA instead of taking arcs iteratively with high probability from one of the parents in the OGA, the philosophy of both crossover operators is the same.

To generate an offspring S_1, an arc of one of the parents will be selected with high probability (similar to crossover). But it is also possible to include new information since all the arcs that allow the creation of a valid tour participate in the roulette with probability greater than 0 (similar to mutation). The value $O/2$ is used because there are two parents, and then $w_{max} = O + 1$ can be interpreted as the maximum weight an arc can have in the roulette (when the arc belongs to both parents). When the arc does not belong to

any parent, it obtains the minimum weight w_{min} in the roulette, that is $w_{min} = 1$. Then, O determines the relative weight between crossover and mutation.

Formally, while visiting city i, the probability of choosing an arc (i, j) to generate the offspring S_1 is defined by equation (1).

$$\mathcal{P}_{ij} = \begin{cases} \frac{w_{ij}}{\sum_{\forall h \in \mathcal{N}_i} w_{ih}} & \text{if } j \in \mathcal{N}_i. \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Next, the main pseudocode of the OGA is presented.

Pseudocode of the *Omicron Genetic Algorithm (version 1)*

```
g = 0                                        /* Initialization of the generation counter */
P = Initialize population ()                 /* Random generation of p individuals */
REPEAT UNTIL end condition
  REPEAT p TIMES
    {F1, F2} = Reproduction (P)              /* Selection of 2 parents through a roulette */
    S1 = Crossover-Mutation ({F1, F2}, 1)    /* Generation of 1 offspring using {F1, F2} */
    P = Update population (S1)               /* S1 replaces the oldest individual of P */
  g = g + 1                                  /* Increment of the generation counter */
```

The above pseudocode was designed to emphasize the similarity with the SGA. However, the next pseudocode is simpler and completely equivalent.

Pseudocode of the *Omicron Genetic Algorithm (version 2)*

```
u = 0                                        /* Initialization of the iteration counter */
P = Initialize population ()                 /* Random generation of p individuals */
REPEAT UNTIL end condition
  {F1, F2} = Reproduction (P)                /* Selection of 2 parents through a roulette */
  S1 = Crossover-Mutation ({F1, F2}, 1)      /* Generation of 1 offspring using {F1, F2} */
  P = Update population (S1)                 /* S1 replaces the oldest individual of P */
  u = u + 1                                  /* Increment of the iteration counter */
```

3.2.4 Example of an OGA's Iteration

To clarify the previous procedure, an example considering the TSP with 5 cities mentioned above is presented next. $O = 4$ and $p = 4$ are considered for this case.

- **Reproduction**

The example assumes an initial population $P = \{P_x\}$ composed of 4 randomly selected individuals with their respective fitnesses f_x. This initial population is presented next.

First randomly chosen individual: $P_1 = \{c1, c4, c3, c2, c5\}$ with $f_1 = 10$

Second randomly chosen individual: $P_2 = \{c1, c3, c2, c5, c4\}$ with $f_2 = 8$

Third randomly chosen individual: $P_3 = \{c3, c5, c1, c2, c4\}$ with $f_3 = 1$

Fourth randomly chosen individual: $P_4 = \{c2, c5, c4, c1, c3\}$ with $f_4 = 5$

Two parents are randomly selected through a roulette, where the weights of the individuals in the roulette are their fitness. It is assumed that individuals P_1 and P_4 are selected to be parents.

$F_1 = \{c1, c4, c3, c2, c5\} = \{(c1, c4), (c4, c3), (c3, c2), (c2, c5), (c5, c1)\}$

$F_2 = \{c2, c5, c4, c1, c3\} = \{(c2, c5), (c5, c4), (c4, c1), (c1, c3), (c3, c2)\}$

- **Crossover-Mutation. Iteration 1**

First, an initial city is randomly chosen to perform Crossover-Mutation. $c4$ is assumed as the initial city. Then, $\mathcal{N}_{c4}$ is composed by $\{c1, c2, c3, c5\}$, i.e. the set of not yet visited cities. The arc $(c4, c2)$ has a weight of 1 in the roulette because it does not belong to any parent. Arcs $\{(c4, c3), (c4, c5)\}$ have a weight of $1 + \frac{O}{2} = 3$ in the roulette because they belong to one parent. Finally, the arc $(c4, c1)$ has a weight of $1 + O = 5$ in the roulette because it belongs to both parents. It is assumed that the arc $(c4, c3)$ is randomly chosen through the roulette.

- **Crossover-Mutation. Iteration 2**

$\mathcal{N}_{c3}$ is composed by $\{c1, c2, c5\}$. The arc $(c3, c5)$ has a weight of 1 in the roulette because it does not belong to any parent. The arc $(c3, c1)$ has a weight of $1 + \frac{O}{2} = 3$ in the roulette because it belongs to one parent. Finally, the arc $(c3, c2)$ has a weight of $1 + O = 5$ in the roulette because it belongs to both parents. It is assumed that the arc $(c3, c2)$ is randomly chosen through the roulette.

- **Crossover-Mutation. Iteration 3**

$\mathcal{N}_{c2}$ is composed by $\{c1, c5\}$. The arc $(c2, c1)$ has a weight of 1 in the roulette because it does not belong to any parent. Finally, the arc $(c2, c5)$ has a weight of $1 + O = 5$ in the roulette because it belongs to both parents. It is assumed that the arc $(c2, c1)$ is randomly chosen through the roulette.

- **Crossover-Mutation. Iteration 4**

$\mathcal{N}_{c1}$ is composed by $\{c5\}$. The arc $(c1, c5)$ has a weight of $1 + \frac{O}{2} = 3$ in the roulette because it belongs to one parent. The arc $(c1, c5)$ is chosen because it is the unique arc represented in the roulette.

In conclusion, the new offspring is $S_1 = \{c4, c3, c2, c1, c5\} = \{(c4, c3), (c3, c2), (c2, c1), (c1, c5), (c5, c4)\}$. Notice that S_1 has 3 arcs of F_1 $\{(c4, c3), (c3, c2), (c1, c5)\}$ and 2 arcs of F_2 $\{(c3, c2), (c1, c5)\}$. Also, S_1 has an arc $\{(c2, c1)\}$ that does not belong to any parent. This shows that the objectives of the operators (crossover and mutation) have not been altered.

- **Population Update**

The new individual S_1 replaces the oldest individual P_1. Next, the new population is shown.
$P_1 = \{c4, c3, c2, c1, c5\}$ with $f_1 = 7$
$P_2 = \{c1, c3, c2, c5, c4\}$ with $f_2 = 8$
$P_3 = \{c3, c5, c1, c2, c4\}$ with $f_3 = 1$
$P_4 = \{c2, c5, c4, c1, c3\}$ with $f_4 = 5$
The entire procedure above is done iteratively until an end condition is satisfied.
Note that OGA is another version of GA like many other published versions.

4 New Strategies for OGA

New strategies referred to crossover, population update and heuristic information are proposed for OGA. Considering that the most relevant aspect mentioned above is the crossover of multiple parents, this new version is called Multi-Parent OGA (MOGA).

4.1 Crossover

This Section considers the generation of the offspring through the crossover of multiple parents. This idea is not new and it was proposed before [10]. More specifically, this strategy proposes that the p individuals of P are parents without any roulette intervention. Obviously, this crossover of p parents eliminates competition among individuals during reproduction. Nevertheless, the new population update strategy proposed in the next Section will solve this competition problem. Considering that there are p parents instead of 2, a weight of O/p is added to each arc (i, j) belonging to every F_x, i.e. $w_{ij} = w_{ij} + O/p \ \forall (i, j) \in F_x$. This way, when an arc belongs to the p parents, the weight of the arc will be $w_{max} = O + 1$. When an arc does not belong to any parent, the weight of the arc will be $w_{min} = 1$. This is done to maintain the weight limits (w_{max} and w_{min}).

4.2 Population Update

To reduce the possibilities that a bad individual enters the population, a competition strategy among offspring is considered. This new strategy replaces the most traditional parent competition strategy. This strategy consists on the generation of t offspring $\{S_1, ..., S_t\}$ in one iteration. Only the offspring with the best fitness ($S_{best} \in \{S_1, ..., S_t\}$) is chosen to enter P. As in the OGA population update strategy, S_{best} replaces the oldest individual of the population. Notice that the same effect is obtained (competition among individuals) with a different strategy.

4.3 Heuristic Information

Good TSP tours are composed with high probability by arcs with short length. Thus, it seems a good idea to give them better weights in the roulette. Then, considering the heuristic information $\eta_{ij} = 1/d(i,j)$, the probability of choosing an arc (i,j) to generate the offspring S_1 is now defined by equation (2).

$$\mathcal{P}_{ij} = \begin{cases} \frac{w_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{\forall h \in \mathcal{N}_i} w_{ih}^{\alpha} \cdot \eta_{ih}^{\beta}} & \text{if } j \in \mathcal{N}_i. \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

Where the input parameters α and β are defined as the relative influence between the weight of the genetic information and the heuristic information η.

Next, the main pseudocode of the MOGA is presented.

Pseudocode of the *Multi-Parent Omicron Genetic Algorithm*

```
u = 0                                  /* Initialization of the iteration counter */
P = Initialize population ()           /* Random generation of p individuals */
REPEAT UNTIL end condition
  S = Crossover-Mutation (P, t)        /* Generation of t offspring using P as parents */
  S_best = best element of S
  P = Update population (S_best)       /* S_best replaces the oldest individual of P */
  u = u + 1                            /* Increment of the iteration counter */
```

So far several versions of GAs have been analyzed. Before the relationship of both algorithms is presented, ACO versions are considered in the next Section.

5 Ant Colony Optimization

Ant Colony Optimization (ACO) is a metaheuristic inspired by the behavior of ant colonies [2]. In the last few years, ACO has received increased attention by the scientific community as can be seen by the growing number of publications and its different fields of application [11]. Even though there exist several ACO variants, the one that may be considered a standard approach is presented next [6].

5.1 Standard Approach

ACO uses a pheromone matrix $\tau = \{\tau_{ij}\}$ for the construction of potential good solutions. The initial values of τ are set $\tau_{ij} = \tau_{init}\ \forall(i,j)$, where $\tau_{init} > 0$. It also takes advantage of heuristic information using $\eta_{ij} = 1/d(i,j)$. Parameters α and β define the relative influence between the heuristic information and the pheromone levels. While visiting city i, $\mathcal{N}_i$ represents the set of cities not yet visited and the probability of choosing a city j at city i is defined as

$$\mathcal{P}_{ij} = \begin{cases} \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{\forall h \in \mathcal{N}_i} \tau_{ih}^{\alpha} \cdot \eta_{ih}^{\beta}} & \text{if } j \in \mathcal{N}_i \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

At every generation of the algorithm, each of the m ants constructs a complete tour using (3), starting at a randomly chosen city. Pheromone evaporation is applied for all (i,j) according to $\tau_{ij} = (1-\rho) \cdot \tau_{ij}$, where parameter $\rho \in (0,1]$ determines the evaporation rate. Considering an elitist strategy, the best solution found so far r_{best} updates τ according to $\tau_{ij} = \tau_{ij} + \Delta\tau$, where $\Delta\tau = 1/l(r_{best})$ if $(i,j) \in r_{best}$ and $\Delta\tau = 0$ if $(i,j) \notin r_{best}$. For one of the best performing ACO algorithms, the $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System ($\mathcal{MMAS}$) [11], minimum and maximum values are imposed to τ (τ_{min} and τ_{max}).

5.2 Population-based ACO

The Population-based ACOs (P-ACOs) were designed by Guntsch and Middendorf [6] for dynamic combinatorial optimization problems. As the standard approach, the initial values of τ are set $\tau_{ij} = \tau_{init}\ \forall(i,j)$, where $\tau_{init} > 0$. The P-ACO approach updates in a different way the pheromone information than the standard approach. P-ACO derives the pheromone matrix through a population $Q = \{Q_x\}$ of q good solutions or individuals as follows. First, at every iteration each of the m ants constructs a solution using probabilities given in equation (3), the best solution enters the population Q. Whenever a solution Q_{in} enters the

population, then τ_{ij} is updated according to $\tau_{ij} = \tau_{ij} + \Delta\tau$, where $\Delta\tau = \Delta$ if $(i,j) \in Q_{in}$ and $\Delta\tau = 0$ if $(i,j) \notin Q_{in}$. After the first q solutions enter Q, i.e. the initialization of the population is finished, one solution Q_{out} must leave the population at every iteration. The solution that must leave the population is decided by an update strategy. Whenever a solution Q_{out} leaves the population, then $\tau_{ij} = \tau_{ij} - \Delta\tau$, where $\Delta\tau = \Delta$ if $(i,j) \in Q_{out}$ and $\Delta\tau = 0$ if $(i,j) \notin Q_{out}$. P-ACO replaces the pheromone evaporation used by the standard approach in this way. The value Δ is a constant determined by the following input parameters, size of the population q, minimum or initial pheromone level τ_{init} and maximum pheromone level τ_{max}. Thus, $\Delta = (\tau_{max} - \tau_{init})/q$ [6].

5.2.1 FIFO-Queue Update Strategy

The FIFO-Queue update strategy was the first P-ACO strategy designed [6], trying to simulate the behavior of the standard approach of ACO. In the FIFO-Queue update strategy, Q_{out} is the oldest individual of Q.

Next, the main pseudocode of the P-ACO FIFO-Queue is presented.

Pseudocode of the *P-ACO FIFO-Queue*

```
u = 0                                   /* Initialization of the iteration counter */
Q = Initialize population ()                  /* Generation of q individuals */
REPEAT UNTIL end condition
  τ = Update pheromone matrix (Q)                    /* Update τ using Q */
  T = Construct solutions (τ)              /* Generation of m solutions using τ */
  T_best = best solution of T
  Q = Update population (T_best)        /* T_best replaces the oldest individual of Q */
  u = u + 1                              /* Increment of the iteration counter */
```

5.2.2 Quality Update Strategy

A variety of strategies was studied in [7], one of them is the *Quality* update strategy. The worst solution (considering quality) of the set $\{Q, Q_{in}\}$ leaves the population in this strategy. This ensures that the best solutions found so far make up the population.

Next, the main pseudocode of the P-ACO *Quality* is presented.

Pseudocode of the *P-ACO Quality*

```
u = 0                                   /* Initialization of the iteration counter */
Q = Initialize population ()                  /* Generation of q individuals */
REPEAT UNTIL end condition
  τ = Update pheromone matrix (Q)                    /* Update τ using Q */
  T = Construct solutions (τ)              /* Generation of m solutions using τ */
  T_best = best solution of T
  Q = Update population (T_best)     /* T_best replaces the worst individual of {Q, T_best} */
  u = u + 1                              /* Increment of the iteration counter */
```

5.3 Omicron ACO

In the search for a new ACO analytical tool, Omicron ACO (OA) was developed [4]. OA was inspired by $\mathcal{MMAS}$, an elitist ACO currently considered among the best performing algorithms for the TSP [11]. It is based on the hypothesis that it is convenient to search nearby good solutions [11].

The main difference between $\mathcal{MMAS}$ and OA is the way the algorithms update the pheromone matrix. In OA, a constant pheromone matrix τ^0 with $\tau^0_{ij} = 1, \forall i,j$ is defined. OA maintains a population $Q = \{Q_x\}$ of q individuals or solutions, the best unique ones found so far. The best individual of Q at any moment is called Q^*, while the worst individual Q_{worst}.

In OA the first population is chosen using τ^0. At every iteration a new individual Q_{new} is generated, replacing $Q_{worst} \in Q$ if Q_{new} is better than Q_{worst} and different from any other $Q_x \in Q$. After K iterations, τ is recalculated using the input parameter Omicron (O). First, $\tau = \tau^0$; then, O/q is added to each element τ_{ij} for each time an arc (i,j) appears in any of the q individuals present in Q. The above process is repeated every K iterations until the end condition is reached. Note that $1 \leq \tau_{ij} \leq (1+O)$, where $\tau_{ij} = 1$ if arc (i,j) is not present in any Q_x, while $\tau_{ij} = (1+O)$ if arc (i,j) is in every Q_x.

Even considering their different origins, OA results similar to the Population-based ACO algorithms described above [6, 7]. The main difference between the OA and the *Quality* Strategy of P-ACO is that OA does not allow identical individuals in its population. Also, OA updates τ every K iterations, while P-ACO updates τ every iteration.

Next, the main pseudocode of the OA is presented.

Pseudocode of the *OA*

```
u = 0                                        /* Initialization of the iteration counter */
Q = Initialize population ()                 /* Generation of q individuals */
REPEAT UNTIL end condition
  τ = Update pheromone matrix (Q)            /* Update τ using Q */
  REPEAT K TIMES
    Q_new = Construct a solution (τ)         /* Generation of 1 solution using τ */
    IF Q_new ∉ Q
      Q = Update population (Q_new)          /* Q_new replaces the worst individual of {Q, Q_new} */
    u = u + 1                                /* Increment of the iteration counter */
```

6 Relationship between GAs and ACO Algorithms

Considering a P-ACO FIFO-Queue and a MOGA with the same population size (i.e. $q = p$), the same number of ants or offspring (i.e. $m = t$) and the same τ_{ij}, w_{ij} limits (i.e. $\tau_{max} = O + 1$ and $\tau_{init} = 1$), the amount of pheromones an ant deposits and the genetic weight of an individual are the same also (i.e. $(\tau_{max} - \tau_{init})/q = O/p$). Paying attention to the pheromone matrix update and the roulette generation explained before, it is easy to see now that both procedures are identical. Besides, the solution construction procedures and probabilities for both algorithms, shown by equations (2) and (3), are the same.

Although the initialization of the population of the P-ACO FIFO-Queue is not entirely at random as the MOGA, this aspect is irrelevant for these algorithms main functionality. Consequently, both algorithms are the same as it is shown in the next pseudocodes, where the Crossover-Mutation is divided in two stages.

Pseudocode of the *Multi-Parent Omicron Genetic Algorithm*

```
u = 0                                        /* Initialization of the iteration counter */
P = Initialize population ()                 /* Random generation of p individuals */
REPEAT UNTIL end condition
  R = Generate roulette (P)                  /* Generation of the roulette using P as parents */
  S = Crossover-Mutation (R, t)              /* Generation of t offspring through the roulette */
  S_best = best element of S
  P = Update population (S_best)             /* S_best replaces the oldest individual of P */
  u = u + 1                                  /* Increment of the iteration counter */
```

Pseudocode of the *P-ACO FIFO-Queue*

```
u = 0                                        /* Initialization of the iteration counter */
Q = Initialize population ()                 /* Generation of q individuals */
REPEAT UNTIL end condition
  τ = Update pheromone matrix (Q)            /* Update τ using Q */
  T = Construct solutions (τ)                /* Generation of m solutions using τ */
  T_best = best solution of T
  Q = Update population (T_best)             /* T_best replaces the oldest individual of Q */
  u = u + 1                                  /* Increment of the iteration counter */
```

6.1 New Strategy for MOGA

In this Section a new population update strategy for MOGA is proposed. Consider that S_{best} replaces the worst element of the set $\{P, S_{best}\}$ instead of the oldest individual, i.e. if $fitness(S_{best})$ is greater than $fitness(P_{worst})$ then S_{best} replaces P_{worst} in P, where P_{worst} is the worst individual of P. This new strategy can be seen as a super-elitism due to the fact that P is composed by the best individuals found so far. Thus, this version is called Super-Elitist MOGA (EMOGA).

At this point it is easy to conclude that EMOGA is equivalent to the *Quality* version of P-ACO. Main pseudocodes of EMOGA and P-ACO *Quality* are presented next to reinforce the above conclusion.

Pseudocode of the *Super-Elitist Multi-Parent Omicron Genetic Algorithm*

```
u = 0                                        /* Initialization of the iteration counter */
P = Initialize population ()                 /* Random generation of p individuals */
REPEAT UNTIL end condition
  R = Generate roulette (P)                  /* Generation of the roulette using P as parents */
  S = Crossover-Mutation (R, t)              /* Generation of t offspring through the roulette */
  S_best = best element of S
  P = Update population (S_best)             /* S_best replaces the worst individual of {P, S_best} */
  u = u + 1                                  /* Increment of the iteration counter */
```

Pseudocode of the *P-ACO Quality*

```
u = 0                                        /* Initialization of the iteration counter */
Q = Initialize population ()                 /* Generation of q individuals */
REPEAT UNTIL end condition
  τ = Update pheromone matrix (Q)            /* Update τ using Q */
  T = Construct solutions (τ)                /* Generation of m solutions using τ */
  T_best = best solution of T
  Q = Update population (T_best)             /* T_best replaces the worst individual of {Q, T_best} */
  u = u + 1                                  /* Increment of the iteration counter */
```

6.2 New Strategy for EMOGA

An undesired characteristic of EMOGA is that one very good individual can dominate the population. In this way, the genetic diversity, a desirable property of GAs, is strongly reduced. A simple way to avoid this problem is to modify the population update strategy. So, in this new strategy S_{best} replaces P_{worst} only if it is different to the whole population. Due to the competition among individuals imposed by this population update strategy, the offspring competition is useless and will be eliminated. Besides, because of the little dynamism caused by this population update strategy, the roulette will be generated every H iterations without significant consequences.

This new version is called Diversified EMOGA (DEMOGA) because it forces genetic diversification. The main pseudocode of the DEMOGA and OA are presented next to show that both algorithms are similar.

Pseudocode of the *Diversified Super-Elitist Multi-Parent Omicron Genetic Algorithm*

```
u = 0                                        /* Initialization of the iteration counter */
P = Initialize population ()                 /* Random generation of p individuals */
REPEAT UNTIL end condition
  R = Generate roulette (P)                  /* Generation of the roulette using P as parents */
  REPEAT H TIMES
    S_1 = Crossover-Mutation (R, 1)          /* Generation of 1 offspring using the roulette */
    IF S_1 ∉ P
      P = Update population (S_1)            /* S_1 replaces the worst individual of {P, S_1} */
    u = u + 1                                /* Increment of the iteration counter */
```

Pseudocode of the *OA*

```
u = 0                                        /* Initialization of the iteration counter */
Q = Initialize population ()                 /* Generation of q individuals */
REPEAT UNTIL end condition
  τ = Update pheromone matrix (Q)            /* Update τ using Q */
  REPEAT K TIMES
    Q_new = Construct a solution (τ)         /* Generation of 1 solution using τ */
    IF Q_new ∉ Q
      Q = Update population (Q_new)          /* Q_new replaces the worst individual of {Q, Q_new} */
    u = u + 1                                /* Increment of the iteration counter */
```

To clarify the origins and equivalences of so many versions, a summarized chart is presented in Figure 1.

7 ACO Algorithms and GAs in TSP

The reasons of ACO's success were studied in [4]. This Section briefly introduces these reasons and relates them with the reasons of GAs' success.

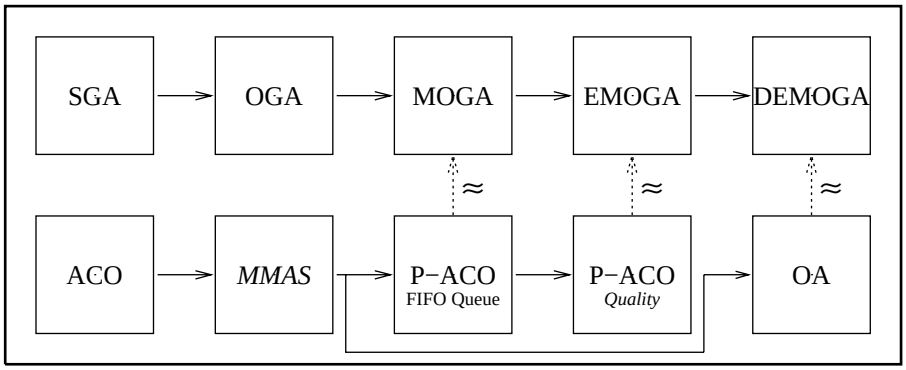


Figure 1: Summarized chart for the versions of GAs and ACO algorithms

7.1 Reasons of ACO's Success in TSP

Previous works found for different TSP instances, a positive correlation between the length of a tour $l(r_x)$ and its distance to the optimal solution $\delta(r_x, r^*)$ [1, 4, 11]. Consequently, the TSP solution space has a globally convex structure [9].

To understand the typical behavior of ACO, the n-dimensional TSP search space is simplified to two dimensions for a geometrical view in Figure 2. A population $Q1 = \{Q1_i\}$ of good solutions uniformly distributed is assumed in Figure 2. Considering that OA gives more pheromones to the good solutions $Q1_i$ already found, this can be seen as a search made close to each $Q1_i$. Thus, OA concentrates the search for new solutions in a central zone of $Q1$, denoted as Ω_{Q1}, which is the zone close to all $Q1_i$. Then OA typically replaces the worst solution of $Q1$ ($Q1_{worst}$) by a new solution Q_{new} of smaller length. A new population $Q2$ is created including Q_{new}. This is shown in Figure 2 with a dotted line arrow. As a consequence, it is expected that the mean distance of $Q2$ to r^* is shorter than the mean distance of $Q1$ to r^* because of the positive correlation mentioned above, i.e. it is expected that the subspace where the search of potential solutions is concentrated decreases. OA performs this procedure repeatedly to decrease the search zone where promising solutions are located. Considering population $Qz = \{Qz_i\}$ for $z >> 2$, Figure 2 shows how Ω_{Qz} has decreased considerably as a consequence of the globally convex structure of the TSP solution space.

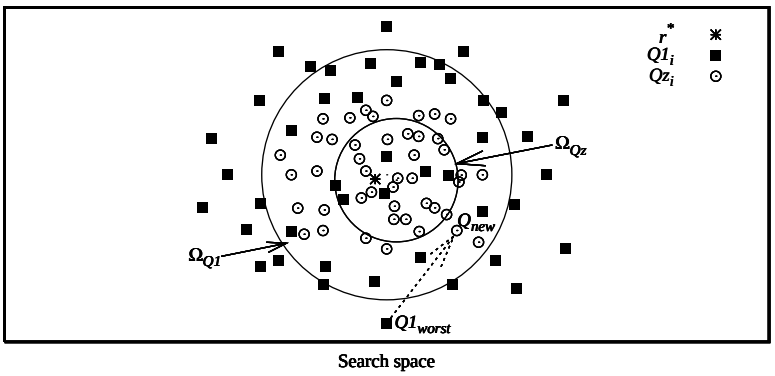


Figure 2: Simplified view of OA behavior

7.2 Reasons of GAs' Success in TSP

As OA and DEMOGA are similar algorithms, the same reasons for success in TSP can be concluded for both algorithms. Therefore, it can be inferred that ACO algorithms and GAs use similar principles to obtain their good performances shown in combinatorial optimization problems.

OA concentrates the search in the central zone of its whole population, where promising solutions are located in a globally convex structure. OGA concentrates the search in the central zone of two individuals of a population through the crossover. Even so, the principle is the same and it explains the important role of the crossover in globally convex structures.

If a good individual P_{good} dominates a population P in OGA, the crossover loses its ability to search in the central zone of P. Thus, the OGA's search is concentrated in a zone close to P_{good} and this population-

based algorithm behaves as a one point search algorithm, like local search. When few individuals dominate a population, the genetic diversity is reduced. Consequently, the loss of genetic diversity eliminates the benefits of the crossover and the search is done mainly through the mutation.

8 Conclusions and Future Work

This work shows that a Genetic Algorithm (MOGA), designed similarly to many others in the literature, is the same algorithm as a Population-based ACO called P-ACO FIFO-Queue. So, GAs and ACO use the same principles to succeed in the TSP (a combinatorial optimization problem), that has a globally convex structure of its solution space.

Besides, this paper explains that Omicron ACO is a Genetic Algorithm with multiple parents, super elitism and forced genetic diversity. Then, the reasons of OA's success presented in [4] can be extended to the GAs. Finally, through this extension, it can be explained the significance of the crossover and the genetic diversity, i.e. they allow the search in the central zone of a population, where promising solutions are located in globally convex structures.

Future work should study the solution space structure of different combinatorial optimization problems and identify those with globally convex structures, where these algorithms have shown to be very successful.

References

[1] Kenneth D. Boese. Cost Versus Distance in the Traveling Salesman Problem. Technical Report 950018, University of California, Los Angeles, Computer Science, May 19, 1995.

[2] Marco Dorigo and Gianni Di Caro. The Ant Colony Optimization Meta-Heuristic. In David Corne, Marco Dorigo, and Fred Glover, editors, *New Ideas in Optimization*, pages 11–32. McGraw-Hill, London, 1999.

[3] Marco Dorigo and Thomas Stützle. The Ant Colony Optimization Metaheuristic: Algorithms, Applications, and Advances. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*, volume 57 of *International Series in Operations Research and Management Science*. Kluwer Academic Publishers, 2003.

[4] Osvaldo Gómez and Benjamín Barán. Reasons of ACO's Success in TSP. In M. Dorigo, M. Birattari, C. Blum, L. M. Gambardella, F. Mondada, and T. Stützle, editors, *Proceedings of ANTS 2004 - Fourth International Workshop on Ant Colony Optimization and Swarm Intelligence*, volume 3172 of *LNCS*, Brussels, September 2004. Springer-Verlag.

[5] David Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

[6] Michael Guntsch and Martin Middendorf. A Population Based Approach for ACO. In Stefano Cagnoni, Jens Gottlieb, Emma Hart, Martin Middendorf, and Günther Raidl, editors, *Applications of Evolutionary Computing, Proceedings of EvoWorkshops2002: EvoCOP, EvoIASP, EvoSTim*, volume 2279, pages 71–80, Kinsale, Ireland, 3-4 2002. Springer-Verlag.

[7] Michael Guntsch and Martin Middendorf. Applying Population Based ACO to Dynamic Optimization Problems. In *Ant Algorithms, Proceedings of Third International Workshop ANTS 2002*, volume 2463 of *LNCS*, pages 111–122, 2002.

[8] John H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, 1975.

[9] T. C. Hu, Victor Klee, and David Larman. Optimization of globally convex functions. *SIAM Journal on Control and Optimization*, 27(5):1026–1047, September 1989.

[10] Heinz Mühlenbein and Hans-Michael Voigt. Gene Pool Recombination in Genetic Algorithms. In Ibrahim H. Osman and James P. Kelly, editors, *Proceedings of the Meta-heuristics Conference*, pages 53–62, Norwell, USA, 1995. Kluwer Academic Publishers.

[11] Thomas Stützle and Holger H. Hoos. $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System. *Future Generation Computer Systems*, 16(8):889–914, June 2000.

Huya: un Sistema para Recuperación de Imágenes basado en MRML

Robinson S. Rivas-Suárez, Yeny Hernández
Universidad Central de Venezuela, Escuela de Computación
Centro de Computación Paralela y Distribuida
Caracas 1040, Venezuela
{rrivas,yhernand}@strix.ciens.ucv.ve

Resumen

En la actualidad existen numerosos sistemas que permiten recuperar imágenes en base al contenido de las mismas, llamados genéricamente Content-Based Image Retrieval (CBIR). Algunos de estos sistemas se basan en la arquitectura Cliente/Servidor, sin embargo no existen estándares aceptados por la comunidad de investigación para comunicar diferentes sistemas CBIR Cliente/Servidor. Para resolver este problema, se introdujo el protocolo MRML, basado en XML y diseñado para permitir la comunicación entre diferentes sistemas CBIR independientemente de los algoritmos, estructuras de datos y tecnologías utilizadas para su implementación.

En este trabajo, se diseñó e implementó un servidor MRML utilizando tecnologías portables, de acuerdo a la especificación 1.0 del lenguaje. Sin embargo esta especificación no resuelve todas las complejidades de la interacción de los sistemas CBIR, por lo que en este trabajo se definieron algunas extensiones a dicha especificación. Finalmente se presenta el sistema desarrollado así como pruebas de su funcionalidad, y propuestas para nuevas extensiones del protocolo.

Palabras clave: MRML, Recuperación de Imágenes Basada en Contenido, XML, Sistemas Basados en WEB

Abstract

Actually there are many systems that search on Images Databases based on the image's content or meaning, usually referred to as Content Based Image Retrieval systems (CBIR systems). Some of these CBIR systems are designed using a Client/Server architecture, so it is in theory possible to communicate different kinds of clients and servers once they share the minimum protocols. However, there is not a well accepted standard protocol to satisfy that objective. To achieve this goal it was proposed MRML as an open XML specification useful for any CBIR system based on Client/Server architecture.

Since the first specification (MRML version 1.0) did not fulfill all the complexities of web-based CBIR systems, we proposed some new ideas and extensions for the protocol, such as indexes vectors specifications, both local and remote images search and sessions managing. Some of these new ideas were implemented and tested in a Java MRML server, the first as far as we know. In this work we present the extensions proposed for MRML and a brief discussion about the differences of our proposals and those of MRML specification team.

Key Words: MRML, Content-Based Image Retrieval, XML, Web Based Systems

1. Introducción

Los sistemas de búsqueda de imágenes basada en contenido (CBIR - Content Based Image Retrieval) se han popularizado debido principalmente al crecimiento de la Internet, lo que ha puesto a disposición de la comunidad académica y empresarial millones de imágenes en los más diversos ámbitos de aplicación. La mayoría de los sistemas que se han creado utilizan algoritmos, técnicas de búsqueda, estructuras de datos y modelos de Bases de Datos propietarios, lo que ha dificultado la interoperabilidad entre los mismos, y por tanto la adecuada validación de las técnicas utilizadas y la interacción entre grupos de investigación y desarrollo **[1, 4]**.

En vista de esta situación, diversos autores han presentado propuestas para definir un protocolo que permita la interacción entre los sistemas CBIR disponibles en la Internet. Uno de estos protocolos es el MRML (Multimedia Retrieval Markup Language) definido y presentado por Henning Müeller y Stéphane Marchand-Maillet **[8]**. Este protocolo se basa en XML y permite definir la mayoría de las interacciones necesarias para establecer comunicación, definir requerimientos y obtener respuestas desde servidores de objetos multimedia, particularmente imágenes. La idea principal es que se puedan independizar los diferentes elementos de los que consta un sistema CBIR Cliente/Servidor en una intranet o basado en Web.

Sin embargo, no son muchos los desarrollos que se han hecho basados en este protocolo. Aparte de las especificaciones y pruebas de los autores **[8,9]** y del proyecto GIFT de la Free Software Foundation **[3]** no existen clientes y servidores comerciales basados en MRML. En este trabajo, se presenta el sistema HUYA desarrollado en la Universidad Central de Venezuela, que consta de un servidor MRML y un cliente de prueba, que permite implementar las principales funcionalidades del protocolo en su versión 1.0. Durante el desarrollo del sistema, se vio la necesidad de aumentar las funcionalidades del protocolo, por lo que se han hecho propuestas en esa dirección. En particular, la especificación implementada añade operaciones para la transmisión de *vectores característicos* de imágenes y la búsqueda en repositorios de imágenes tanto locales como remotos.

El sistema desarrollado se implementó usando Java y herramientas de software abierto, y puede ser ejecutada en cualquier máquina donde haya una JVM. Se hicieron pruebas de funcionamiento sobre colecciones de imágenes indexadas manualmente y automáticamente, y sobre colecciones de imágenes locales y remotas al cliente, usando varias estructuras de datos multidimensionales para la indexación de las imágenes.

El trabajo consta de cinco secciones, incluyendo esta introducción. En la sección 2 se discuten las características principales de los sistemas CBIR, en cuanto sistemas de almacenamiento, indexación y búsqueda de información multimedia. En la sección 3 se muestra una panorámica del lenguaje MRML, sus principales características y las especificaciones de la versión 1.0. Igualmente se presentan algunas de las modificaciones que se hicieron a la especificación.

La sección 4 muestra la arquitectura del sistema desarrollado y el ambiente de desarrollo y prueba. Por último, la sección 5 presenta los resultados y conclusiones obtenidos hasta el momento en este proyecto, así como una perspectiva de los trabajos en desarrollo y futuros.

2. Sistemas CBIR

Por sistemas CBIR se entiende una serie de tecnologías que permiten al usuario almacenar, indexar, clasificar y posteriormente recuperar información de imágenes digitales **[4]**. Debido a que los sistemas CBIR se encargan de varias actividades anteriores y posteriores a la recuperación de las imágenes, es común caracterizarlos como sistemas modulares donde cada uno de los módulos se encarga de una o varias primitivas de procesamiento, entre las que destacan:

1.- **Extraer características de las imágenes:** Lo que se hace de forma automática o semi-automática, de manera que las características sirvan para generar índices que faciliten la posterior clasificación y búsqueda.
2.- **Almacenar los índices:** para lo cual se utilizan Estructuras de Datos y manejadores de Bases de Datos especializados para el manejo de información multidimensional, que es como usualmente se define la información proveniente de imágenes y otros objetos Multimedia.
3.- **Construir las solicitudes de búsqueda:** de acuerdo a un patrón o patrones proporcionados por el usuario o sistema cliente. Por lo general al tratarse de imágenes, se utiliza el paradigma llamado

Búsqueda por Muestra o QBE (Query by Example) en la que el usuario o sistema cliente presenta una imagen que sirve de modelo para la búsqueda, y el sistema retorna las imágenes más parecidas de acuerdo a las métricas asociadas a las estructuras de datos y algoritmos especializados.
4.- **Retornar los resultados del proceso de búsqueda:** de acuerdo a especificaciones definidas por el motor de búsqueda o por el cliente.

Puede observarse que los diferentes pasos en la construcción y funcionamiento de los sistemas CBIR no indican nada acerca de la arquitectura del sistema. Se puede conceptualizar dichos sistemas como sistemas centralizados donde los diferentes pasos de funcionamiento los ejecuta un solo sistema modular, o como sistemas distribuidos en los que cada operación la realiza un elemento distribuido.

Para cada uno de los pasos descritos anteriormente se han desarrollado numerosas técnicas, algoritmos y arquitecturas como se describirá brevemente de inmediato.

1.1. Modelos de Extracción de Características

Las características que se pueden extraer de una imagen se pueden clasificar en varias categorías de acuerdo al nivel de abstracción y a las semánticas asociadas a las características mismas. Según Eakins **[1]** normalmente se definen tres niveles de extracción de características de imágenes:

- **Características Primitivas**: son aquellas basadas exclusivamente en la información numérica asociada a la imagen digital, por ejemplo brillo, luminosidad, etc.
- **Características Lógicas**: son aquellas que representan elementos geométricos, morfológicos o pictóricos que requieren de la aplicación de técnicas especializadas sobre la información numérica, por ejemplo, la detección de rostros, la presencia de ciertas formas o figuras dentro de una imagen, etc.
- **Características Abstractas**: Son aquellas que representan valores abstractos de la realidad. En general, dependen de técnicas muy avanzadas relacionadas con Inteligencia Artificial, por ejemplo la detección de emociones humanas, categorías políticas, deportivas, etc.

La extracción de características se puede hacer de forma manual, semi-automática o automática. En **[15, 6]** se presentan ejemplos de sistemas que emplean diferentes tipos de técnicas para la extracción de características.

1.2.- Estructuras de Datos Multidimensionales

Una vez extraídas las características de las imágenes, éstas deben ser codificadas de acuerdo a algun criterio y almacenadas en estructuras de datos que permitan su rápida búsqueda y manipulación. En general, de cada imagen se obtienen múltiples valores que representan en su conjunto una coordenada de un espacio N-dimensional. Muchas estructuras de datos se han definido para almacenar esta información multidimensional, tales como R-Trees, Q-Trees, SR-Trees, KD-Trees, etc.

Las diferencias entre las estructuras de datos vienen dadas por su capacidad para almacenar distintos tipos de información, almacenar información redundante o incompleta, hacer búsquedas por regiones, etc. Las estructuras de datos están asociadas además a los algoritmos requeridos para realizar las consultas, en especial los llamados algoritmos de búsqueda de los vecinos más próximos *(nearest neighbor searching).* Estos algoritmos permiten que, dada una imagen de referencia, se extraigan de la estructura de datos aquellas imágenes que más se asemejen a la referenciada de acuerdo a algunas métricas.

1.3. Construcción de las Solicitudes

Se han definido algunos lenguajes para indicar cómo un cliente debe especificar lo que desea buscar en un ambiente de recuperación de información multimedia, específicamente imágenes. Aun cuando a la fecha no se tiene un estándar universalmente aceptado, sí se han presentado diferentes propuestas para construir dichas solicitudes **[2]**. Algunas propuestas manejan lenguajes de consulta textuales como MMDOC-QL **[7]**. Otras se basan en la construcción de prototipos de imágenes como base de la búsqueda como QBIC de IBM **[11]**, mientras que otros se basan en la presentación de una o más imágenes de referencia, usando la técnica conocida como Búsqueda por Muestra (*Query by Example*) **[6,10,15]**.

La forma como se especifican las características del objeto a consultar es independiente de la forma como se realiza

dicha consulta, ya que las técnicas de búsqueda dependen de los algoritmos asociados a las Estructuras de Datos de almacenamiento. En general, en ambientes Cliente/Servidor, la especificación de la búsqueda se encuentra del lado del cliente, mientras que los algoritmos de búsqueda y las estructuras de almacenamiento se encuentran del lado del servidor.

1.4.- Retorno de los resultados

La última fase se limita a la presentación de las búsquedas efectuadas por el usuario. La forma, número de resultados, niveles de relevancia, información adicional que se presente a los clientes y otras consideraciones dependen de las especificaciones de las interfaces de usuario, y están en correspondencia directa con las capacidades de los algoritmos de búsqueda, las estructuras de almacenamiento y los lenguajes de especificación.

Los pasos explicados anteriormente se efectúan independientemente de la arquitectura de software, ya sea centralizada, distribuida o Cliente/Servidor. Sin embargo, en el caso de sistemas Cliente/Servidor (ver figura 2) se hace necesario independizar los el funcionamiento de los módulos Clientes y Servidores mediante el uso de un protocolo de comunicaciones que permita la interacción de diferentes clientes con diferentes servidores sin importar el lenguaje de programación, el Sistema Operativo, las plataformas de hardware, etc, tal como se tiene para plataformas maduras basadas en web (protocolos http, ftp o telnet). En la sección siguiente se muestra una propuesta funcional para lograr este objetivo.

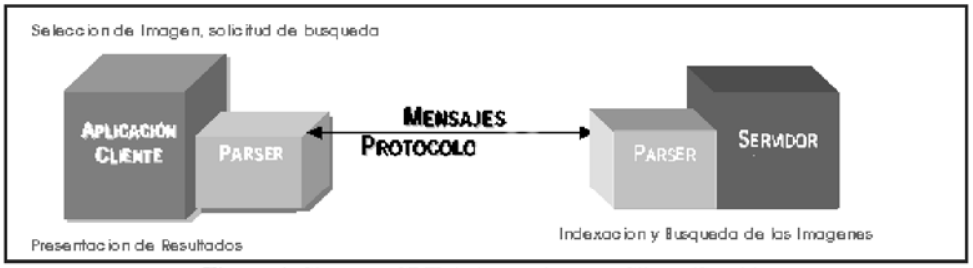


Figura 1: Sistemas CBIR bajo arquitectura Cliente/Servidor

3. El lenguaje MRML

El lenguaje de marcado para recuperación multimedia MRML (*Multimedia Retrieval Markup Language*) fue diseñado por Wolfgang Müller, Henning Müller y Stéphane Marchand-Meillet de la Universidad de Ginebra **[8]** como una forma de independizar los clientes y servidores en sistemas CBIR. MRML está basado en XML y define las principales interacciones que se requieren para establecer comunicaciones entre dos módulos del sistema CBIR, entre ellas establecer conexión, iniciar sesiones, obtener información sobre capacidades del servidor, enviar requerimientos al servidor y retornar respuestas al cliente. Un diagrama de estados de MRML se muestra en la figura 2.

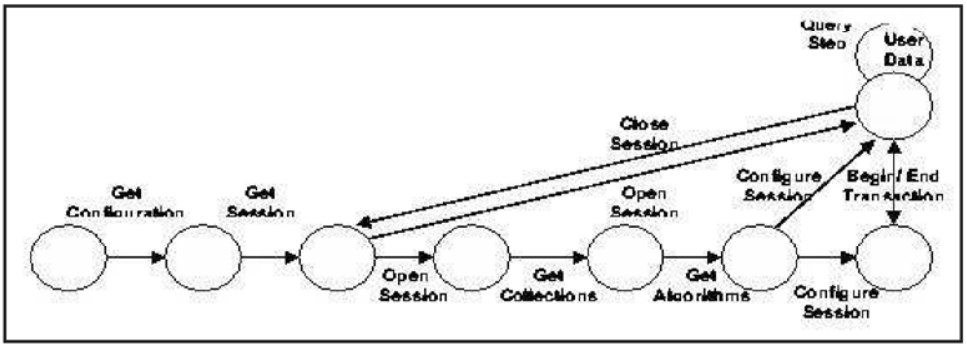


Figura 2: Diagrama de Transición de Estados de MRML

La comunicación Cliente-Servidor en MRML es una secuencia de conexiones. En cada conexión una petición o un conjunto de peticiones son respondidas usando un mensaje simple o un pequeño grupo de mensajes. La máquina de estados describe la comunicación, comenzando en el punto donde se hace el primer contacto con el servidor.

El cliente establece el primer contacto con el servidor enviando el mensaje de solicitud *get-server-properties*. Como respuesta el cliente recibe un mensaje *server-properties*, que en la especificación MRML 1.0 se define como vacío. Sin embargo este mensaje es importante si se piensa en la extensibilidad que ofrece el MRML para desarrollos futuros. Después de recibir la información de la descripción del servidor el cliente podrá enviar una petición para obtener una lista de las sesiones del usuario, usando la etiqueta *get-sessions*.

El mensaje de respuesta a esto es el *session-list*. El cliente podrá ahora abrir una sesión a través del mensaje *open-session*, obteniendo un *acknowledge-session-op* como respuesta. La sesión abierta es requerida para tener control sobre el manejo de estados, para poder realizar posteriores consultas. Abriendo la sesión, el servidor habrá recibido la identificación de usuario, la contraseña y la identificación del usuario, de existir en el sistema.

Todo lo anteriormente expuesto se necesita para saber qué colecciones y qué algoritmos el usuario puede ver. Después de obtener un identificador de sesión y una sesión abierta el cliente tiene la posibilidad de consultar tanto las colecciones como los algoritmos disponibles para cada colección. Las formas de consulta sobre los elementos anteriormente nombrados son dependientes del paradigma de consulta establecido. En particular un algoritmo puede contener un atributo como un identificador que lo una al tipo de colección en que puede ser usado (por ejemplo, algoritmos de similaridad de rostros solo podrían ser usados sobre colecciones de imágenes de rostros). Teniendo acceso a la lista de algoritmos y colecciones, el cliente tiene acceso a suficiente información para poder configurar la sesión que ya ha sido establecida.

Después de esto la sesión está totalmente configurada y pueden efectuarse las consultas (mediante el mensaje *query-step*). Las consultas pueden ser agrupadas dentro de un grupo de transacciones. El cliente también debe ser capaz de poder enviar los datos de usuario *user-data* al servidor. Estas etiquetas con datos del usuario contienen información sobre el usuario y se especifican para propósitos de aprendizaje por parte del servidor.

Al solicitar una consulta al servidor, la respuesta es un mensaje *query-result*. La especificación original MRML indica que los objetos de la colección se especifican mediante un URL, de manera que la localización real de los objetos puede ser arbitraria. Sin embargo, esta simplificación también significa que los clientes no pueden enciar las características de los objetos que desean consultar al servidor sino únicamente los URL, lo que hace que todo el trabajo de indexación y extracción de características recaiga sobre el servidor una vez dispone de los URL de consulta.

Los mensajes MRML 1.0 se especifican mediante un DTD XML donde se detallan los tipos de mensajes y la construcción semántica de MRML. En la figura 3 se muestra un extracto del DTD de MRML en el que se muestran los tipos de mensaje existentes. Obsérvese que no hay especificaciones de seguridad, búsquedas locales o remotas, mantenimiento de colecciones, etc.

```
<!ELEMENT mrml (begin-transaction?,
  (get-configuration
  |configuration-description
  |get-sessions
  |session-list
  |open-session
  |rename-session
  |close-session
  |delete-session
  |get-collections
  |collection-list
  |get-algorithms
  |algorithm-list
  |configure-session
  |query-step
  |query-result
  |user-data
  |error)?,
  end-transaction?)>
<!ATTLIST mrml
   session-id CDATA #IMPLIED
   transaction-id CDATA #IMPLIED
   mrml-id ID #IMPLIED>
```

Figura 3: Extracto del DTD de MRML 1.0, con los tipos demensaje soportados

El DTD, la especificación formal de MRML 1.0 y documentación al respecto se encuentran en el sitio oficial de

MRML (http://www.mrml.net). Dos ejemplos de mensaje MRML se muestran en las figuras 4 y 5.

```
<!-- OPEN-SESSION: ESTABLECE COMUNICACIÓN CON EL SERVIDOR
-->
<?XML VERSION="1.0" ENCODING="UTF-8"?>
<!DOCTYPE MRML SYSTEM "HTTP://150.185.75.106/HUYA/MRML.DTD">
<MRML SESSION-ID="2003-10-09 20:23:30.494">
<OPEN-SESSION USER-NAME="YENYCAROL" SESSION-NAME="OPEN-SESSION" />
</MRML>
```

Figura 4: Mensaje de establecimiento de sesión.

```
<?XML VERSION="1.0" ENCODING="UTF-8"?>
<!DOCTYPE MRML SYSTEM "HTTP://150.185.75.106/HUYA/MRML.DTD">
<MRML SESSION-ID="2003-10-09 20:23:30.494" >
<GET-COLLECTIONS>
<COLLECTION
COLLECTION-LOCATION-STRING="HTTP://150.185.75.106/HUYA/AMUSEMENT/"/>
</GET-COLLECTIONS>
</MRML>
```

Figura 5: Mensaje de solicitud de colecciones

4. El Sistema HUYA

En base a la versión 1.0 de MRML, se inició en la Escuela de Computación de la Universidad Central de Venezuela la construcción de un sistema Cliente/Servidor basado en Java que sirviera de plataforma de prueba para el protocolo. El sistema independiza las operaciones y definiciones del Cliente y el Servidor, que solamente interactúan por medio del pase de mensajes MRML y operan contra colecciones de datos que se acceden a través de un servidor Web, que puede estar independiente tanto del cliente como del servidor. El nombre del sistema se colocó como momenaje a un planeta extra-solar descubierto por un astrofísico Venezolano durante el año 2003. La figura 6 muestra la arquitectura del sistema Huya.

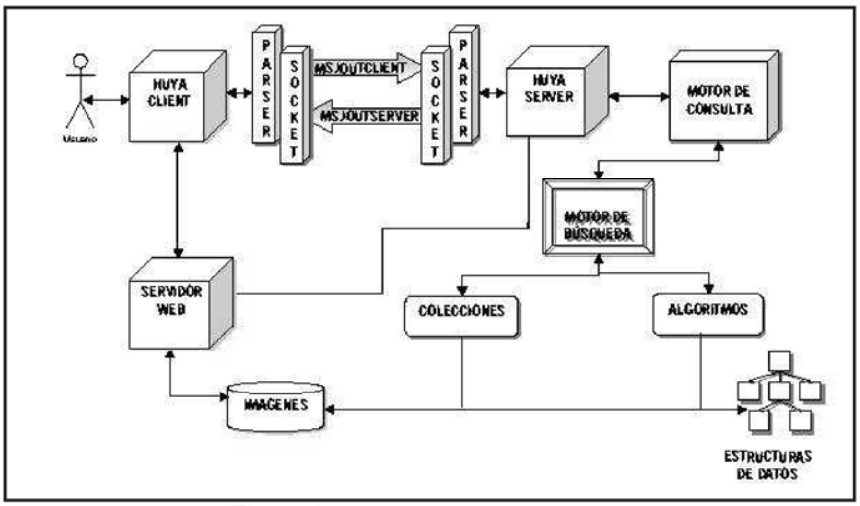


Figura 6: Arquitectura del Sistema Huya

4.1. Esquema de Interacción Cliente/Servidor

El sistema Huya se basa en el modelo solicitud-respuesta, en el que cada mensaje es tratado como una conexión individual, de manera que no hay una conexión abierta permanente entre el Cliente y el Servidor. Los pasos de la interacción del sistema se detallan a continuación:

1.- **Obtener Configuración:** Inicialmente el cliente establece una conexión con el servidor MRML en escucha, al establecer la conexión, el Cliente requiere conocer las propiedades de búsqueda y las colecciones de imágenes que posee el Servidor.

2.- **Enviar la Descripción de la Configuración:** El Servidor responde con las propiedades establecidas en su archivo de configuración, de manera tal que envía un mensaje al Cliente, con los Algoritmos de Búsqueda que posee y las Colecciones de Imágenes de respuesta. Las imágenes se encuentran físicamente en repositorios accesibles mediante el Servidor Web.

3.- **Obtener Imágenes de una Colección:** Si el Cliente desea conocer las imágenes de muestra de una colección dada, envía al servidor un mensaje, indicando el nombre de la colección seleccionada

3.1.- **Obtener los URL's de las Imágenes de la Colección:** Dado el nombre de una colección, el servidor consulta al servidor web y construye una lista con los URL's de las imágenes contenidas en la colección.

3.2.- **Retornar la Lista con las Imágenes de la Colección:** Esta actividad muestra la interacción realizada entre el servidor HuyaServer y el servidor web. En ese momento el servidor posee una lista de URL's de todas las imágenes de muestra.

3.3.-**Enviar la Lista al Cliente:** La lista de los URL's de las imágenes de muestra, es enviada al Cliente.

3.4.-**Solicitar las Imágenes al Servidor Web:** Una vez obtenida la lista de los URL's de las imágenes de muestra, el Cliente realiza una petición al Servidor web.

3.5.- **Retornar Imágenes**: El Servidor web retorna las imágenes consultadas por el Cliente.

4.- **Iniciar las Consultas:** Una vez obtenidas las imágenes de muestra, el cliente está en capacidad de realizar una o más consultas al servidor HuyaServer, en cualquiera de las dos formas predefinidas (ver sección 4.2).

4.1.- **Si el cliente desea realizar una consulta sobre una imagen local**, éste debe poseer el vector de características de la imagen seleccionada, de manera tal que pueda enviar este vector al servidor para que ejecute la consulta. Junto con el vector característico de la imagen seleccionada, se envía el nombre del algoritmo de búsqueda y la cantidad de imágenes que requiere como resultado.

4.1.1.- **Consultar las Estructuras de Datos Multidimensionales:** Dado el vector de características de la imagen, se hace un llamado al Algoritmo de Búsqueda seleccionado por el cliente que resuelve la búsqueda generando tantas imágenes resultado como el cliente haya solicitado.

4.2.- **Enviar el URL de la Imagen en caso que la búsqueda sea Remota:** Si el cliente desea realizar una consulta sobre una imagen de una de las colecciones del servidor, simplemente selecciona el URL de la imagen y se envía un mensaje de consulta junto con el nombre de un algoritmo de búsqueda y la cantidad de imágenes que requiere como resultado.

4.2.1.- **Buscar el Vector Característico y Consultar las Estructuras de Datos:** Si el cliente seleccionó una búsqueda remota (imagen de una colección del servidor), el servidor debe ser capaz de buscar el Vector Característico de la imagen de muestra y realizar el correspondiente llamado al Algoritmo de Búsqueda seleccionado por el Cliente.

5.- **Obtener Resultados de las Estructuras de Datos:** La estructura de datos y el Algoritmo seleccionados devuelven una lista con todas las imágenes resultado, esta lista es utilizada por el Servidor para enviar respuesta al Cliente.

6.- **Enviar la Lista de URL de Imágenes:** Una vez obtenidas las imágenes de respuesta, se envía una lista de los URL's de dichas imágenes al Cliente.

7.- **Consultar al Servidor Web:** Una vez el Cliente obtiene la lista de los URL's de las imágenes resultado, se realiza una petición al Servidor web para su presentación al usuario.

8.- **Retornar Imágenes y Mostrar Resultados:** El Servidor web retorna las imágenes consultadas por el Cliente, y éste hace la presentación al usuario.

4.2. Extensiones a MRML 1.0

Durante el desarrollo del proyecto e observó que las interacciones de los sistemas CBIR son más complejas que los mensajes disponibles en la especificación 1.0, por lo que se decidió añadir algunas observaciones al protocolo original. Estas observaciones han sido discutidas con los autores de MRML, que están actualmente definiendo el *draft* de la versión 2.0.

En particular, se vió la necesidad de implementar mensajes para diferenciar las búsquedas *locales* de las búsquedas *remotas*. La diferencia básica entre estos dos tipos de búsqueda es que las búsquedas *locales* permiten al cliente el envío del *vector característico* de una imagen de alguna colección local al cliente, en lugar de enviar el URL de la imagen. La ventaja de esto radica en que se pueden utilizar algoritmos de extracción de características diferentes a los definidos en el lado del servidor, e incluso pueden ser vectores característicos de imágenes sintéticas generadas por el usuario. El query remoto implica comparar imágenes localizadas en las colecciones del lado del servidor, identificándolas mediante su URL. En cualquiera de los dos casos, la comparación se hace sobre los algoritmos y colecciones ofertados por el servidor.

Los nuevos mensajes definidos se ejemplifican mediante las figuras 7 y 8. En ellas se muestra la especificación de los nuevos mensajes *query-local* y *query-remote.*

```
<!-- QUERY_LOCAL: ENVIAR LOS PARAMETROS DE CONSULTA SOBRE UNA
IMAGEN LOCAL SELECCIONADA
-->
<?XML VERSION="1.0" ENCODING="UTF-8"?>
<!DOCTYPE MRML SYSTEM "HTTP://150.185.75.106/HUYA/MRML.DTD">
<MRML SESSION-ID="2003-10-09 20:23:30.494" >
<QUERY-LOCAL>
<IMAGE NAME-IMAGE="AMU001.JPG" IMAGE-VECTOR="20;23;30;49;4"/>
<CANT-IMAGE NUMBER="4" />
<ALGORITHM ALGORITHM-NAME-STRING="RTREE" />
</QUERY-LOCAL>
</MRML>
```

Figura 7: Mensaje *query-local*

```
<!-- QUERY_REMOTE: ENVIAR LOS PARAMETROS DE CONSULTA SOBRE UNA
IMAGEN DE UNA COLECCIÓN SELECCIONADA
-->
<?XML VERSION="1.0" ENCODING="UTF-8"?>
<!DOCTYPE MRML SYSTEM "HTTP://150.185.75.106/HUYA/MRML.DTD">
<MRML SESSION ID="2003-10-09 20:23:30.494" >
<QUERY-REMOTE>
<IMAGE IMAGE-LOCATION-
STRING="HTTP://150.185.75.106/HUYA/BOATS/BOAT01.JPG"/>
<CANT-IMAGE NUMBER="6" />
<ALGORITHM ALGORITHM-NAME-STRING="RTREE" />
<COLLECTION COLLECTION-NAME-STRING="BOATS" />
</QUERY-REMOTE>
</MRML>
```

Figura 8: Mensaje *query-remote*

Obsérvese que en el mensaje *query-local* se envía un vector característico llamado *image-vector* que se obtiene mediante la aplicación de algún algoritmo de extracción de características del lado del cliente. En el caso del mensaje *query-remote*, solamente se envía el URL de la imagen. En ambos casos, la colección donde se buscará la imagen, y el algoritmo de búsqueda se toman de las listas proporcionadas previamente por el Servidor.

4.3. Interfaces de Usuario y Detalles de Funcionamiento del Sistema Huya

A continuación se muestran las Interfaces de Usuario del Sistema Huya. En la figura 9, se muestra la consola del Servidor. Aunque este tipo de consola no es estrictamente parte del Servidor, es importante para efectos de depuración del sistema, y para tener un mejor control de la herramienta.

Figura 9: Consola de Administración de HuyaServer

La intefaz principal del Cliente se muestra en la figura 10. Obsérvese que antes de iniciar la interacción con el Servidor Huya, el Cliente tiene acceso a una o más colecciones de imágenes locales, que obtiene desde un archivo

de configuración XML. Para la implementación del sistema se utilizó el puerto IP 1251, aunque esto no es parte del estándar del protocolo y puede utilizarse cualquiera disponible.

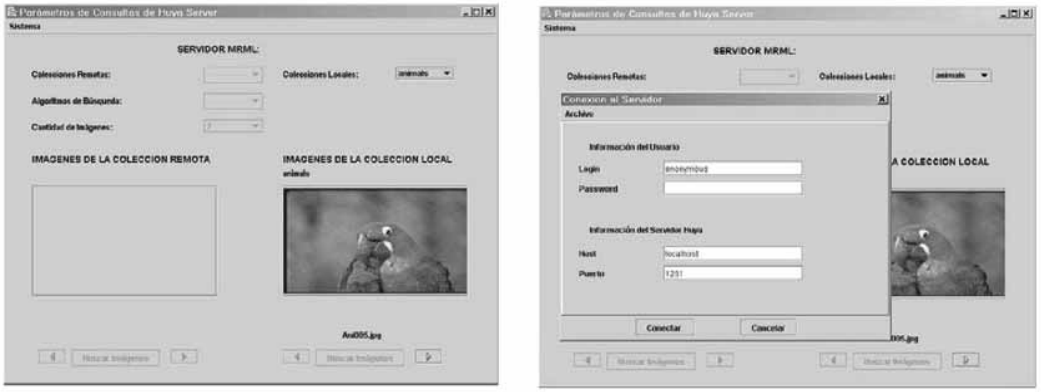


Figura 10: Interfaz de Usuario del Cliente Huya y Ventana de Conexión

Las imágenes de las colecciones de prueba varían en cuanto a temática y formato, incluyendo imágenes de animales, paisajes, botes, flores y escenas de moda. Los vectores característicos se extrajeron de forma semiautomática, e incluyen características pictóricas y de forma (color, brillo, cantidad de objetos, disposición de los objetos, etc). Una descripción de estas características se encuentra en **[13,14]**

Una vez selecionada una forma de interacción, el usuario puede buscar imágenes de las dos formas predeterminadas (Búsqueda Local y Búsqueda Remota). La similitud de las imágenes depende del Algoritmo de Búsqueda Seleccionado, en los experimentos realizados se utilizó la Distancia Euclidiana sobre los vectores multidimensionales, almacenando los vectores característicos en un KDTree. En las figuras 11 y 12 se muestra un ejemplo de resultado obtenido en cada uno de estos tipos de búsqueda.

Figura 11: Resultado de una Búsqueda Local.

Figura 12: Resultado de una Búsqueda Remota

Es importante destacar que el sistema se programó utilizando Interfaces de Java, para lo cual se definió un API conciso que incluye:

- **IndexTreeInterface:** Interfaz para las Estructuras Multidimensionales soportadas por el Servidor.
- **FeatureExtractorInterface:** Interfaz que define el comportamiento de cualquier Algoritmo de Extracción de Caracterísitcas soportado tanto por el Cliente como por el Servidor.
- **FeatureInterface:** Interfaz que define cómo se estructuran las características de cada imagen.
- **CollectionInterface:** Interfaz que permite definir el comportamiento de las Colecciones de Imágenes, y permite encapsular la creación y manipulación de colecciones enteras de imágenes
- **RepositoryInterface:** Permite modelar el comportamiento de repositorios de imágenes. Cada repositorio consiste de: 1) una Estructura Multidimensional para el almacenamiento de los Vectores Característicos de las imágenes, 2) un Algoritmo de Extracción de dichos Vectores Característicos y 3) una Colección de Imágenes.

De esta forma se logró un sistema modular, en el que se pueden añadir nuevos Algoritmos de Extracción de Características y Estructuras de Datos Multidimendionales si se desea, siempre que se programe en Java respetando la Interfaz correspondiente. Para que estas nuevas implementaciones tengan efecto en el Cliente o el Servidor, basta añadirlos dentro del archivo de configuración XML correspondiente. En la figura 13 se muestra un ejemplo de archivo de configuración correspondiente al Servidor.

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
  <name>Huya-Server</name>
  <description>Servidor MRML de la UCV</description>
  <port>1251</port>
  <image-directory>http://ip:port/</image-directory>
  <dtd-url>http://ip:port/dtd/mrml1.0.dtd</dtd-url>
  <collections>
     <collection>
        <collection-name>animals</collection-name>
        <collection-id>0</collection-id>
        <collection-location>http://ip:port/name</collection-location>
     </collection>
  </collections>
  <algorithms>
     <algorithm>
        <algorithm-name>kdtree</algorithm-name>
        <algorithm-id>id_kdtree</algorithm-id>
        <algorithm-type>type_kdtree</algorithm-type>
        <algorithm-location>mrml.kdtree.KDTree</algorithm-location>
     </algorithm>
  <extractors>
     <extractor>
        <extractor-name>Simple Feature Extractor</extractor-name>
        <extractor-location>mrml.api.SimpleFeatureExtractor</extractor-location>
     </extractor>
  </extractors>
</configuration>
```

Figura 13: Ejemplo de Archivo de Configuración del Servidor

Una descripción completa del trabajo preliminar de Huya se encuentra en **[5,14]**.

5. Resultados y Trabajos Futuros

El trabajo realizado permitió obtener los siguientes resultados:

1. Se implementó la especificación 1.0 de MRML de forma modular, de manera que permite interactuar un Cliente contra un Servidor de forma transparente en cuanto a los detalles internos de cada uno de los componentes.
2. Como limitación del trabajo, no se implementaron los mensajes correspondietnes a *user-relevance*, esto es, mensajes diseñados específicamente para permitir que el Sevidor pueda aprender de la interacción del Cliente. Este tipo de aprendizaje está presente en sistemas como PicSOM **[10]** en el cual de las sucesivas búsquedas del Cliente se puede inferir el tipo de respuesta esperada.
3. La especificación de MRML permite la interacción sin la definición explítcita de sesiones, por lo cual éstas no fueron implementadas.
4. Se definieron tres nuevos nuevos mensajes:
   4.1 *collection-list*: permite obtener la lista de todas las imágenes de una colección específica.
   4.2 *query-remote:* permite solicitar resultados en base a una imagen remota, esto es, definida en las colecciones del servidor. Equivale al mensaje *query-step* de la especificación 1.0
   4.3 *query-local*: permite solicitar resultados en base a un Vector Característico provisto por el Cliente, y obtenido de algún algoritmo de extracción de características.

   Los nuevos mensajes se implementaron respetando la estructura sintáctica de MRML, y para su implementación se hizo necesario redefinir parcialmente el DTD de MRML 1.0.
5. La implementación mediante el uso de Java Interfaces ha permitido tener un software modular y fácil de actualizar, ya que los principales componentes del sistema pueden ser añadidos mediante su simple inclusión en archivos de configuración.

Durante el desarrollo del trabajo la interacción con el grupo original que definió MRML permitió encontrar algunas inconsistencias de la especificación original, lo que ha permitido buscar nuevas alternativas en función de la nueva

especificación MRML 2.0 . Actualmente se trabaja para definir nuevas funcionalidades tanto al protocolo como al sistema, entre las que destacan:

En cuanto al protocolo:

1. Soporte para aspectos de seguridad: manejo de sesiones seguras, encriptación, autenticación a nivel de servidor, cliente y mensajes, etc
2. Soporte para manejo de variables de sesión y uso de configuraciones específicas para cada usuario.
3. Soporte para la definición de Vectores Caracterísitcos y Algoritmos de Extracción de Características.
4. Mejorar las definiciones de API's para las colecciones y Repositorios de Imágenes, de manera que se puedan utilizar colecciones mediante un protocolo bien definido, y no solamente mediante el uso de URL's de cada objeto individual.

En cuanto al Sistema Huya, nuestros trabajos se dirigen a:

1. Implementar el manejo de sesiones y la especificación 2.0 (en cuanto esté oficialmente disponible)
2. Adaptar el sistema para el uso de nuevas Estructuras de Datos Multidimensionales, y nueos tipos de Objetos Multimedia como audio y video. Actualmente, algunos avances en este sentido se realizan bajo la dirección del Dr. Joshua Reiss de la universidad Queen Mary en Londres, Inglaterra **[12]**.
3. Relizar experimentación con el sistema GIFT a fin de determinar la portabilidad real de lis sistemas y del protocolo en sí.
4. Definir Repositorios de Imágenes Distribuidos mediante el uso de plataformas abiertas.

Los códigos fuente de la versión actual de Huya, así como documentación adicional, están disponibles en el sitio del proyecto Huya **[14]**

Agradecimientos

Este trabajo se ha desarrollado con la colaboración de varios estudiantes de la Licenciatura en Computación de la Universidad Central de Venezuela. La consola de administración es producto del trabajo de la Br. Marlyn Castro, y los algoritmos de Extraccción de Características usados así como las Estructuras de Datos Multidimensionales fueron programados por Milagors Rengel y Edgar Padilla **[13]** .

Referencias

[1] Eakins J., Graham M.. Content-Based Image Retrieval. *Internal Report Nr. 39, University of Northumbria at Newcastle*, October 1999.

[2] Fernández M., Simeon J., Wadler P., Cluet S., Deutsch A., Florescu D., Levy A., Maier D., McHugh J., Robie J., Suciu D., Widom J., XML query languages: Experiences and exemplars,disponible en *http://www-db.research.belllabs.com/user/simeon/xquery.ps,citeseer.ist.psu.edu/fern99xml.html*,1999

[3] The GIFT (GNU Image Finding Tool) project. http://www.gnu.org/software/gift/gift.html

[4] Goodrum A., Image Information Retrieval: An Overview of Current Research, *Journal of Informing Science, Special Issue on Information Science Research*, vol 3, number 2, 2000.

[5] Hernández Y., Diseño e Implementación de un Servidor MRML, *Trabajo Especial de Grado presentado ante la Universidad Central de Venezuela*, Escuela de Computación, Caracas, Noviembre 2003

[6] Laaksonen J., Koskela M., Laakso S., Oja E.. Self-Organizing Maps as a Relevance Feedback Technique in Content-Based Image Retrieval. *Pattern Analysis & Applications*, vol 4 number 2, pp. 140-152, June 2001.

[7] Liu P.. An Approach to specifiyng and querying multimedia objects and scheduled structures in XML *Documents. Spatial/Temporal Databases Meeting*, Paris 2000.

[8] Müller W. , Müller H., Marchand-Maillet S., et al. MRML: A communication protocol for content-based image retrieval. *International Conference on Visual Information Systems* . Lyon, France,November 2-4, 2000.

[9] Müller W., Pecenovic Z., Müller H., Marchand-Maillet S., Pun T. Et al. MRML: An Extensible Communication Protocol for Interoperability and Benchmarking of Multimedia Information Retrieval Systems. *SPIE Photonics East - Voice, Video, and Data Communications*, Boston, MA, USA, November 5--8, 2000.

[10] PicSOM image Browsing System: sitio web del proyecto PicSOM: http://www.cis.hut.fi/picsom

[11] QBIC: IBM research group. Official Web Site of the Query By Image Content of IBM, http://wwwqbic.almaden.ibm.com

[12] Reiss, Joshua: sitio web del Departamento de Electricidad de la Universidad Queen Mary, Londres,

http://www.elec.qmul.ac.uk, http://www.elec.qmul.ac.uk/~josh

[13] Rengel M., Padilla E., Estudio Comparativo entre Estructuras de Datos Multidimensionales, *Trabajo de Grado presentado ante la Universidad Central de Venezuela*, Escuela de Computación, Caracas, Octubre 2002

[14] UCV Huya: sitio web del proyecto Huya. http://strix.ciens.ucv.ve/~mrml

[15] Wang J. Z., Li J., Wiederhold G.,SIMPLIcity: Semantics-Sensitive Integrated Matching for Picture LIbraries,*IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol 3, number 9, pp 947-963, 2001.

Abordagem para Derivação de Regras de Usabilidade Especializadas em Contextos de Aplicação Específicos

Otávio A Martins Netto
Pontifícia Universidade Católica do Rio de Janeiro
SERG – Departamento de Informática
Rua Marquês de São Vicente, 225 – Gávea – Rio de Janeiro RJ
Brazil – Tel: +55 21 31141500 - ramal 3323
otavionetto@acm.org

Faculdade de Minas
Av. Cristiano Ferreira Varella, 655 – Bairro Universitário – Muriaé MG
Brazil – Tel: +55 32 37297500

Débora Maria Barroso Paiva
Universidade de São Paulo
Departamento de Computação
Av. Trabalhador Sãocarlense, 400 – São Carlos SP
Brazil – Tel: +55 16 33739375
debora@icmc.usp.br

Maria da Graça Pimentel
Universidade de São Paulo
Departamento de Computação
Av. Trabalhador Sãocarlense, 400 – C.P.: 668 – São Carlos SP
Brazil – Tel: +55 16 33739668
mgp@icmc.usp.br

Abstract

Usability principles or rules abound in the literature, particularly general ones that can be applied to many application domains. However, when general rules are applied to specific domains it may be difficult to obtain the best results when compared to rules customized to the domains. We present a method that guides the derivation of rules specialized to a particular domain from the analysis of general rules. We present applications of the method to a domain hypermedia Web; an analysis of the results from experiments using the method presented significant results.

Keywords: Usability rules, Usability, Evaluation.

Resumo

Regras de usabilidade genéricas propostas na literatura são aplicadas em diferentes contextos de sistemas de software. No entanto, a utilização de regras de usabilidade genéricas pode implicar na obtenção de índices de usabilidade não satisfatórios quando comparado aos índices obtidos através do uso de regras de usabilidade especializadas no contexto do sistema. É apresentada neste artigo uma abordagem sistemática, rápida, e de baixo custo para derivação de regras de usabilidade especializadas em contextos de aplicação específicos. Uma aplicação da abordagem proposta no contexto multimídia Web, acompanhada de análises satisfatórias dos resultados obtidos, também é apresentada neste artigo.

Palavras-chave: Regras de usabilidade, Usabilidade, Avaliação.

1. Introdução

Sistemas comerciais, educacionais e de entretenimento estão freqüentes e acessíveis a uma diversidade crescente de usuários. A redução de preço dos computadores, a expansão da World Wide Web e a maior utilização de recursos multimídia são alguns dos principais aspectos que vêm estimulando o uso e a popularização desses sistemas. Neste contexto, é possível observar que a usabilidade ─ qualidade de interação entre usuários e sistemas de software ─ desponta como fator imprescindível ao sucesso de um sistema, uma vez considerada a necessidade de competir com diversos produtos concorrentes e ainda atender a requisitos de diferentes usuários.

Para proporcionar o desenvolvimento de sistemas de boa usabilidade, no entanto, é necessária a execução de atividades de projeto de interfaces durante o processo de desenvolvimento de software.

Algumas atividades componentes de um projeto de interfaces, como por exemplo, atividades de avaliação de usabilidade, geralmente fazem uso de recomendações de projeto denominadas "regras de usabilidade".

O principal objetivo da inserção de regras de usabilidade em atividades de projeto de interfaces consiste em proporcionar aos projetistas mecanismos de orientação durante atividades de identificação de problemas de usabilidade nas interfaces inspecionadas. Além disso, um segundo e não menos importante objetivo do uso de regras de usabilidade consiste em proporcionar mecanismos de auxílio aos projetistas durante tomadas de decisão de projeto, referentes, por exemplo, à forma de apresentação de animações em interfaces de sistemas educacionais (visual ou sonora).

As regras de usabilidade propostas na literatura possuem diferentes níveis de abrangência – definidos em função do escopo de aplicação dessas regras entre projetos e elementos de interfaces distintos.

Neste sentido, as regras de usabilidade podem ser classificadas em duas principais categorias:

- ✓ Regras de Usabilidade Genéricas [5,6,9,11]: Geralmente referenciadas como princípios de usabilidade ou heurísticas, são regras que possuem estruturas de redação generalistas que proporcionam suas aplicações em diversos contextos de sistemas de software. Em outras palavras, regras de usabilidade genéricas podem, por exemplo, ser aplicadas tanto em projetos de interfaces de sistemas multimídia Web quanto em projetos de interfaces de sistemas não multimídia disponíveis em Intranets de acesso restrito a um público bem definido.
- ✓ Regras de Usabilidade Especializadas [10,14,15]: Geralmente referenciadas como guidelines ou padrões, são regras que possuem estruturas de redação especializadas a um contexto específico ou a um conjunto reduzido de sistemas de software. Em outras palavras, regras de usabilidade especializadas podem, por exemplo, ser exclusivamente direcionadas a aspectos de interfaces de sistemas de software não multimídia baseados em formulários.

Em particular, resultados provenientes de [4,7,11,13] revelam que a utilização de regras de usabilidade genéricas pode implicar na obtenção de índices de usabilidade não satisfatórios quando comparado aos índices obtidos através do uso de regras de usabilidade especializadas ao contexto do sistema – especialmente quando envolvidas pessoas não especialistas em usabilidade durante a execução de atividades de projeto de interfaces que fazem uso destas regras.

As regras de usabilidade especializadas a um contexto de aplicação, por sua vez, são geralmente derivadas a partir de observações de padrões de comportamento durante a interação entre usuários e sistemas de software [3,15], assim como a partir de experiências prévias de projetos de interfaces [10,14].

No entanto, essa derivação de regras especializadas a partir de observações da interação entre usuários e sistemas de software contribui com a expansão dos custos de desenvolvimento de um projeto de interfaces, o que não é compatível com a engenharia de baixo custo proposta por Nielsen [9].

Além disso, a derivação de regras especializadas a partir de experiências prévias em projetos de interfaces restringe a capacidade de desenvolvimento de recomendações de projeto à experiência em usabilidade dos projetistas envolvidos.

Desta forma, o objetivo deste artigo consiste na apresentação de uma abordagem sistemática, rápida, e de baixo custo para derivação de regras de usabilidade especializadas em contextos de aplicação específicos, que podem, por exemplo, serem desenvolvidas e posteriormente utilizadas em atividades de projeto de interfaces que envolvem tanto pessoas especialistas quanto pessoas não especialistas em usabilidade.

Esta seção apresentou o contexto no qual se insere este artigo, a motivação para seu desenvolvimento e os resultados alcançados. A segunda seção apresenta conceitos necessários ao entendimento do restante do texto. A terceira seção apresenta a abordagem de derivação de regras de usabilidade especializadas em contextos de aplicação específicos. A quarta seção apresentada um estudo de caso onde regras de usabilidade especializadas no contexto multimídia Web são derivadas a partir da avaliação de dois sistemas pertencentes ao referido contexto. Além disso, a quarta seção apresenta um experimento de validação das regras de usabilidade derivadas. A quinta seção, por sua vez, apresenta as conclusões deste trabalho.

2. Projeto de Interfaces

Métodos de projeto de interfaces, como o Processo de Usabilidade proposto por Nielsen [8], são compostos por um conjunto de atividades complementares de desenvolvimento e de avaliação de usabilidade.

Algumas atividades componentes dos métodos de projeto de interfaces, como por exemplo, as atividades de Avaliação Heurística e Aplicação de Guidelines do Processo de Usabilidade, fazem uso de regras de usabilidade com o intuito de orientar a identificação de problemas de interação entre usuários e sistemas de software.

2.1. Processo e Regras de Usabilidade

A aplicação do Processo de Usabilidade consiste essencialmente na execução iterativa de atividades de desenvolvimento e de avaliação de interfaces – apresentadas na Figura 1.

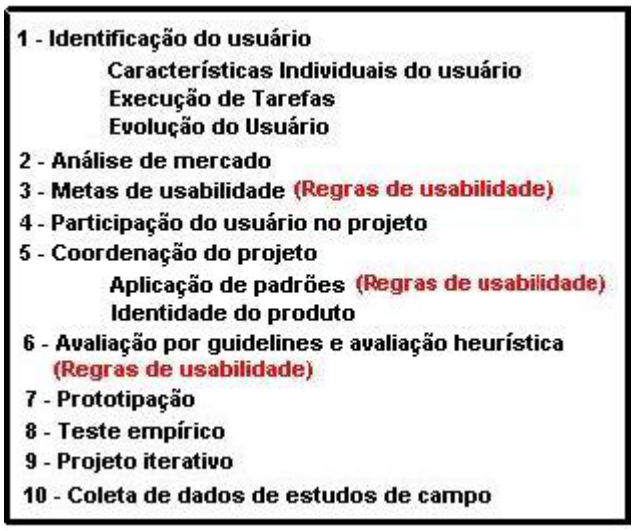


Figura 1 – Atividades componentes do Processo de Usabilidade – Adaptado de[8].

Uma importante atividade componente de diversos métodos de projeto de interfaces é aquela responsável pela seleção de um conjunto de recomendações de projeto – regras de usabilidade genéricas denominadas princípios de usabilidade ou heurísticas – consideradas pelos projetistas de um sistema, desde estágios iniciais de desenvolvimento, como parâmetros componentes da usabilidade daquele produto em particular.

No Processo de Usabilidade de Nielsen essa atividade de seleção, denominada Metas de Usabilidade, ocorre através da análise e posterior classificação de um subconjunto do total de regras de usabilidade genéricas propostas pelo autor [9]. Assim, a partir de resultados provenientes da execução de atividades iniciais do Processo de Usabilidade, como por exemplo, a atividade de Identificação do Usuário, projetistas de interfaces executam análises e seleções de regras de usabilidade genéricas consideradas essenciais à composição da usabilidade de um sistema.

Em suma, as atividades componentes de um método de projeto de interfaces têm por finalidade controlar e analisar a adequação das interfaces de um sistema com as regras de usabilidade genéricas selecionadas pelos projetistas como fundamentais àquele produto em desenvolvimento.

Além das regras de usabilidade genéricas, também podem ser utilizadas em atividades de projeto de interfaces especializações dessas regras em contextos de aplicação específicos, denominadas guidelines ou padrões.

Conforme [5,11] , uma vantagem imediata da utilização de regras de usabilidade especializadas é a maior facilidade de interpretação e conseqüente aplicação das recomendações de projeto transmitidas por elas, principalmente quando envolvidos projetistas não especialistas em usabilidade em atividades de projeto que fazem uso destas regras.

Apesar do Processo de Usabilidade ser composto por um conjunto de atividades de projeto de interfaces, por restrição de tempo e pessoal especializado, somente algumas de suas atividades são freqüentemente executadas durante o processo de desenvolvimento de interfaces de sistemas de software.

Mais especificamente, é possível observar uma maior aplicação da atividade de avaliação de usabilidade denominada Aplicação de Guidelines.

A Aplicação de Guidelines constitui a execução de uma atividade de avaliação de interfaces que faz uso de regras de usabilidade especializadas e organizadas na forma de checklists, que necessitam, no entanto, estar centradas em aspectos específicos do contexto de aplicação do sistema.

No entanto, as regras de usabilidade especializadas a contextos de aplicação específicos propostas na literatura são geralmente derivadas a partir de observações prévias de padrões de comportamento durante a interação entre usuários e sistemas de software [3,15], assim como a partir de experiências prévias em projetos de interfaces [10,14].

3. Abordagem para Derivação de Regras de Usabilidade Especializadas em Contextos de Aplicação Específicos

O objetivo desta seção é apresentar uma abordagem sistemática, rápida e de baixo custo para derivação de regras de usabilidade especializadas em contextos de aplicação específicos.

A abordagem apresentada a seguir, composta por quatro etapas de execução, proporciona a substituição tanto de mecanismos de derivação de recomendações de projeto baseados em observações de comportamento – observações da interação entre usuários e sistemas de software – quanto de mecanismos de derivação baseados em experiências prévias em projetos de interfaces.

A abordagem caracteriza-se por apresentar baixo custo de execução ao eliminar a necessidade de sucessivas e extensas atividades de observação da interação entre usuários e sistemas de software. Além disso, torna a derivação de regras de usabilidade especializadas em contextos de aplicação específicos independente da experiência em usabilidade de projetistas de interfaces em relação ao referido contexto.

Em especial, a abordagem apresentada nesta seção baseia-se na execução de uma técnica de avaliação de usabilidade rápida e barata, denominada Avaliação Heurística, sobre sistemas pertencentes ao contexto para o qual se deseja derivar regras especializadas.

Assim, a partir da análise dos resultados obtidos em experimentos de Avaliação Heurística é possível obter um conjunto de regras de usabilidade especializadas no contexto ao qual pertencem os sistemas analisados.

3.1. Identificação de Problemas de Usabilidade e de Boas Soluções de Projeto

A primeira etapa da abordagem de derivação de regras de usabilidade compreende a execução de experimentos de Avaliação Heurística baseados no uso de regras de usabilidade genéricas (princípios de usabilidade ou heurísticas).

Os experimentos de Avaliação Heurística consistem essencialmente na execução de quatro atividades: planejamento, pré–avaliação, avaliação e pós–avaliação.

A atividade Planejamento compõe-se pela execução das seguintes subatividades:

- ✓ Identificação dos conhecimentos dos potenciais usuários do sistema.

  Estes conhecimentos são identificados em função da experiência dos usuários com o sistema em avaliação, com o contexto de aplicação do sistema e com a área de computação de forma geral.

- ✓ Identificação das regras de usabilidade genéricas componentes da usabilidade do sistema.

  A identificação dessas regras consiste na análise e seleção daquelas consideradas essenciais a uma boa interação entre usuários e sistema. Recomenda-se a utilização da totalidade ou de um subconjunto das regras de usabilidade genéricas propostas por Nielsen [9].

- ✓ Seleção de avaliadores adequados à execução da Avaliação Heurística.

  A seleção de avaliadores envolve a análise de potenciais candidatos priorizando para isso aqueles que possuem experiência em usabilidade de forma geral.

  Vale ressaltar que resultados prévios obtidos em estudos executados no contexto deste trabalho revelaram uma independência da necessidade de utilização de avaliadores com experiência específica no contexto de aplicação do sistema analisado, ou seja, no contexto de aplicação para o qual pretende-se derivar regras especializadas.

- ✓ Preparação de material (software e hardware) necessário à execução do experimento.

  Materiais como software e hardware devem ser preparados antes do início da execução dos experimentos de Avaliação Heurística, ou seja, antes mesmo da chegada dos avaliadores ao ambiente de avaliação.

- ✓ Preparação do ambiente de execução do experimento de Avaliação Heurística.

  É recomendada atenção especial na preparação do ambiente de execução dos experimentos (laboratório, sala, entre outros) de forma a proporcionar condições adequadas e satisfatórias de execução dos mesmos.

Finalizada a atividade Planejamento, inicia-se a execução da atividade Pré–avaliação do experimento de Avaliação Heurística que, por sua vez, compreende a realização das seguintes subatividades:

- ✓ Realização de recomendações referentes à execução de análises nas interfaces inspecionadas. Para isso, deve-se destacar aos avaliadores a necessidade de considerar os conhecimentos dos potenciais usuários do sistema durante as atividades de Avaliação Heurística, e não seus próprios conhecimentos do sistema, de seu domínio de aplicação e em aspectos gerais de computação.

- ✓ Treinamento dos avaliadores referente à aplicação da técnica de Avaliação Heurística, assim como detalhamentos referentes ao escopo de aplicação das regras de usabilidade genéricas selecionadas como componentes da usabilidade do sistema.
- ✓ Elaboração e apresentação aos avaliadores de um roteiro de tarefas a ser seguido durante as inspeções de avaliação de usabilidade.
- ✓ Transmissão aos avaliadores de recomendações adicionais que destacam a necessidade da realização de no mínimo duas inspeções nas interfaces do sistema, a primeira analisando aspectos globais e a segunda analisando elementos específicos da interação entre usuários e sistema.

A terceira atividade do experimento de Avaliação Heurística, denominada Avaliação, é composta pela execução das seguintes subatividades:

- ✓ Realização de sessões de inspeção de usabilidade nas interfaces do sistema. Para isso, os avaliadores devem considerar as informações transmitidas durante a atividade Pré-avaliação, como por exemplo, o roteiro e o conjunto de regras de usabilidade genéricas selecionadas ainda na atividade Planejamento.

  Boas soluções de projeto identificadas nas interfaces avaliadas também devem ser registradas pelos avaliadores durante as sessões de Avaliação Heurística, e não somente os problemas de usabilidade identificados.

  Durante as sessões de inspeção de usabilidade os avaliadores devem ser assistidos por uma pessoa experiente em usabilidade – um experimentador.

  O papel do experimentador restringe-se a orientar os avaliadores na solução de eventuais dúvidas durante as inspeções de usabilidade, atentando para que desta forma não ocorram intervenções nos resultados do experimento.

  Além disso, cuidado adicional deve ser tomado de forma a proporcionar a execução de inspeções de usabilidade individuais, assegurando assim a integridade dos resultados obtidos pelos diferentes avaliadores.
- ✓ Associação dos problemas de usabilidade identificados a pelo menos uma das regras de usabilidade genéricas utilizadas.

  Vale ressaltar a obrigatoriedade de associação de cada um dos problemas de usabilidade identificados a pelo menos uma das regras de usabilidade genéricas utilizadas no experimento.

Por fim, a atividade Pós–avaliação é responsável pela análise dos resultados coletados nos experimentos de Avaliação Heurística. Desta forma, esta atividade compreende a execução das seguintes subatividades:

- ✓ Composição de uma listagem única de problemas de usabilidade identificados pelos avaliadores envolvidos no experimento de Avaliação Heurística. A geração dessa listagem envolve a eliminação de problemas de usabilidade redundantes, ou seja, eliminação de duplicidades de problemas identificados por diferentes avaliadores.
- ✓ Composição de uma listagem única de boas soluções de projeto identificadas pelos avaliadores durante o experimento de Avaliação Heurística.
- ✓ Para as subatividades anteriores, recomenda-se a participação de todos avaliadores envolvidos no experimento – uma vez considerada a necessidade de analisar e posteriormente validar os problemas identificados, seus respectivos relacionamentos com as regras de usabilidade utilizadas no experimento e as boas soluções de projeto também identificadas pelos avaliadores.

3.2. Análise Qualitativa dos Problemas de Usabilidade e das Boas Soluções de Projeto Identificadas

A segunda etapa da abordagem de derivação de regras de usabilidade ocorre após a conclusão dos experimentos de Avaliação Heurística.

Essa etapa consiste essencialmente na análise qualitativa e na organização dos problemas identificados em grupos associados especificamente a elementos de navegação hipertexto (links de navegação, links de estruturas de acesso – menus, mapas de navegação, nomes de URL's, entre outros) ou a mídias de informação isoladas (texto, gráfico, áudio ou vídeo).

Assim, a organização de problemas de usabilidade em elementos de navegação ou em mídias de informação, proporciona melhores mecanismos de análise de validação dos mesmos.

Um problema de usabilidade é considerado válido quando as pessoas envolvidas nesta etapa da abordagem de derivação de regras de usabilidade identificam uma relação concreta com alguma das regras de usabilidade genéricas consideradas como componentes da usabilidade do sistema avaliado (etapa Planejamento), mesmo que a regra de usabilidade associada não seja aquela registrada pelos avaliadores durante o experimento de Avaliação Heurística.

Recomenda-se, baseado em experiências anteriores de execução da abordagem de derivação apresentada, o envolvimento também nesta etapa de todos avaliadores responsáveis pela execução das inspeções de Avaliação Heurística.

3.3. Derivação de Regras de Usabilidade Especializadas

A terceira e última etapa da abordagem de derivação de regras de usabilidade consiste finalmente na produção de um conjunto de regras de usabilidade especializadas no contexto de aplicação do(s) sistema(s) avaliado(s).

Para isso, o experimentador e os avaliadores envolvidos nos experimentos de Avaliação Heurística realizam redações individuais dos problemas de usabilidade registrados originalmente durante as inspeções de usabilidade nas interfaces analisadas.

Além disso, o experimentador e os avaliadores realizam também individualmente redações das boas soluções de projeto registradas durante a execução dos experimentos de avaliação de usabilidade.

As redações das regras de usabilidade especializadas são executadas respeitando-se os agrupamentos dos problemas de usabilidade e das boas soluções de projeto realizados na etapa anterior da abordagem de derivação.

Após as redações individuais são realizadas reuniões entre o experimentador e os avaliadores. Nestas reuniões são realizadas revisões das redações e uma listagem final de regras de usabilidade especializadas é produzida, respeitando-se, no entanto, a organização dessas regras em elementos de navegação hipertexto e em mídias de informação isoladas.

Após a geração da listagem de regras de usabilidade especializadas são realizados experimentos de avaliação de usabilidade baseados em checklists. Esses experimentos baseiam-se na utilização de pessoas não especialistas em usabilidade para, que desta forma, seja analisada a adequação das regras de usabilidade derivadas.

Um conjunto de regras de usabilidade especializadas a um contexto de aplicação é considerado adequado quando, a partir das regras de usabilidade especializadas, avaliadores não especialistas em usabilidade conseguem identificar nas interfaces inspecionadas aproximadamente o mesmo conjunto de problemas de usabilidade identificados por avaliadores experientes ao utilizar a técnica de Avaliação Heurística e as regras de usabilidade genéricas selecionadas.

Para analisar quantitativamente possíveis diferenças entre o número de problemas de usabilidade identificados nos experimentos de Avaliação Heurística originais (baseados em avaliadores especialistas e regras de usabilidade genéricas) e os experimentos baseados em checklists (baseados em avaliadores não especialistas e regras de usabilidade especializadas derivadas), recomenda-se fortemente a utilização de mecanismos de análise estatística.

O teste de Wilcoxon, utilizado nesta etapa da abordagem de derivação, é responsável pela análise estatística da diferença entre dois valores. Esse método foi selecionado por comparar se valores obtidos em dois instantes distintos – número de problemas de usabilidade identificados nos experimentos de Avaliação Heurística e nos experimentos baseados em checklists – são ou não estatisticamente significativos. Além disso, o nível de significância utilizado e considerado aceitável é igual a 0,1.

Mais informações sobre o método estatístico Wilcoxon podem ser obtidas em [16].

4. Estudo de Caso: Abordagem para Derivação de Regras de Usabilidade Especializadas no Contexto Multimídia Web

Esta seção apresenta uma aplicação da abordagem de derivação de regras de usabilidade especializadas em contextos de aplicação específicos. O contexto selecionado para o estudo de caso foi o das aplicações multimídia Web.

Neste sentido, foram selecionadas duas aplicações multimídia Web (iClass – http://coweb.icmc.sc.usp.br/iclass – e Jornal da Lilian – http://www.terra.com.br/jornaldalilian) para avaliação e posterior derivação de regras de usabilidade especializadas naquele contexto.

4.1. Sistema iClass

O sistema iClass [1,2], desenvolvido no *Georgia Institute of Technology* e no Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo, investiga problemas associados ao desenvolvimento de ambientes instrumentados de captura de material multimídia em salas de aula, bem como a apresentação desse material através de hiperdocumentos multimídia na Web [12].

Como ambiente instrumentado é utilizada uma sala como aquela apresentada na Figura 2. A sala é equipada com lousa eletrônica (A), câmera de vídeo (B), microfones (C), projetor apresentando hiperdocumentos Web visitados (D) e projetor apresentando informações demonstradas anteriormente na lousa eletrônica (E).

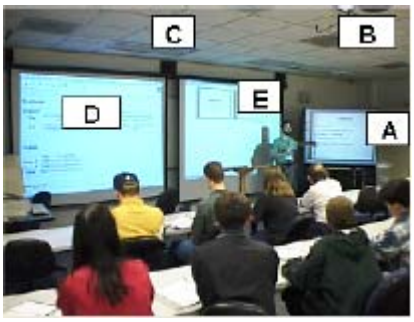


Figura 2 – Sala de aula denominada iClass [1].

Figura 3 – Hiperdocumento típico produzido automaticamente pelo sistema iClass ao término de uma aula [1].

Após a aula, professores e alunos podem ter acesso ao material capturado através de hiperdocumentos multimídia gerados automaticamente pelo sistema. Um exemplo típico de hiperdocumento gerado é apresentado na Figura 3. A janela em segundo plano apresenta o conteúdo da aula como segue: o frame da direita apresenta os slides e as anotações inseridas pelo professor durante a aula, as anotações são ligações que indexam as informações de áudio e vídeo apresentadas no frame inferior esquerdo; o frame superior esquerdo apresenta uma linha do tempo que indexa as informações de mídia dependente de tempo; as marcações à direita da linha do tempo correspondem a ligações para os slides apresentados no frame à direita, ou para hiperdocumentos Web visitados durante a aula e apresentados em janelas separadas quando selecionadas, como àquela em primeiro plano.

4.2. Identificação de Problemas de Usabilidade e de Boas Soluções de Projeto

Seguindo a abordagem de derivação de regras de usabilidade especializadas o experimento de Avaliação Heurística do sistema iClass foi dividido em quatro atividades: Planejamento, Pré–avaliação, Avaliação e Pós–avaliação.

4.2.1. Planejamento

Nesta atividade do experimento de Avaliação Heurística foi executada a subatividade de identificação das características dos potenciais usuários do sistema iClass. Essas características dos usuários consistem em seus conhecimentos sobre o sistema em análise, em seus conhecimentos sobre o contexto de aplicação do sistema (no caso do iClass, o contexto multimídia Web) e em seus conhecimentos acerca de aspectos gerais da área de computação.

Após a identificação dos conhecimentos dos potenciais usuários do sistema foram identificadas, dentre as dez regras de usabilidade genéricas propostas por Nielsen [9], aquelas consideradas componentes da usabilidade do iClass.

Ainda na etapa Planejamento, foram selecionados 5 avaliadores para realizar as inspeções nas interfaces do sistema iClass. A experiência em usabilidade de forma geral, e não necessariamente no contexto multimídia Web, foi um fator decisivo durante a atividade de seleção dos mesmos. A opção por um número de avaliadores igual a 5 é justificada através de recomendações referentes a condução adequada de experimentos de Avaliação Heurística [9].

Dois dos cinco avaliadores selecionados possuíam experiência na execução de atividades de desenvolvimento e de avaliação de usabilidade de sistemas Web não multimídia. Os demais avaliadores selecionados, por sua vez, possuíam conhecimentos em usabilidade adquiridos em trabalhos desenvolvidos na disciplina de Interação Humano-Computador do Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo.

Por fim, na atividade Planejamento foram executadas subatividades de preparação dos equipamentos (software e hardware) necessários à execução do experimento de Avaliação Heurística do sistema iClass.

4.2.2. Pré-avaliação

Durante esta atividade foi apresentado aos avaliadores a descrição dos conhecimentos dos potenciais usuários do sistema iClass mapeados na atividade anterior. Instruções também foram passadas aos avaliadores para que as inspeções nos hiperdocumentos multimídia do sistema iClass fossem realizadas sob a perspectiva dos conhecimentos dos usuários, e não sob a perspectiva dos conhecimentos dos próprios avaliadores.

Em seguida, foram apresentados aos avaliadores a técnica de Avaliação Heurística, as regras de usabilidade genéricas selecionadas (consistência e familiaridade) e um roteiro de tarefas a serem executadas nas interfaces do sistema iClass definido ainda na atividade Planejamento [7].

Recomendações adicionais sobre o procedimento de execução das inspeções de avaliação de usabilidade também foram transmitidas aos avaliadores, como por exemplo, a execução de no mínimo duas inspeções em cada

hiperdocumento do sistema — a primeira analisando aspectos globais e a segunda analisando elementos específicos da interação usuário–sistema.

4.2.3. Avaliação

Nesta atividade do experimento de Avaliação Heurística foram executadas as sessões de inspeção de usabilidade nos hiperdocumentos multimídia do sistema iClass. Para isso, os avaliadores utilizaram as regras de usabilidade genéricas selecionadas na atividade Planejamento e apresentadas a eles na atividade Pré-avaliação.

Um experimentador – pessoa experiente em avaliação de usabilidade – assistiu aos avaliadores durante as inspeções nas interfaces do sistema, auxiliando-os na solução de eventuais dúvidas sobre o iClass atentando para que essas interferências não influenciassem nos resultados finais do experimento de avaliação de usabilidade.

As sessões de avaliação foram executadas individualmente, de forma que um avaliador não pudesse se comunicar ou até mesmo interferir nas inspeções de outros avaliadores.

Os problemas identificados pelos avaliadores foram registrados em papel e associados às regras de usabilidade genéricas (princípios de usabilidade ou heurísticas) utilizadas no experimento.

4.2.4. Pós-avaliação

Nesta etapa o experimentador e os avaliadores envolvidos no experimento de Avaliação Heurística executaram análises qualitativas e quantitativas dos dados coletados na atividade.

A execução de inspeções de usabilidade nas interfaces do sistema iClass consumiu aproximadamente 11 horas e 36 minutos de trabalho. O número médio de sessões de inspeção de usabilidade utilizadas por cada avaliador foi igual a dois e o tempo médio de cada avaliação foi igual a 2 horas e 19 minutos.

Uma listagem final contendo 73 de problemas de usabilidade foi gerada a partir das anotações individuais dos avaliadores, eliminando assim eventuais redundâncias existentes [13].

Durante a geração desta listagem foram analisados todos os potenciais problemas de usabilidade identificados pelos avaliadores. Além disso, foi analisado nesta atividade de pós-avaliação a conformidade dos problemas identificados com as regras de usabilidade relacionadas.

Uma listagem final das boas soluções de projeto identificadas na atividade Avaliação, contendo um total de 19 itens, também foi gerada, complementando assim os aspectos de interação entre usuários e sistema componentes da boa usabilidade no contexto multimídia Web.

4.3. Análise Qualitativa dos Problemas de Usabilidade e das Boas Soluções de Projeto Identificadas

Análises qualitativas sobre os resultados computados na atividade Pós-avaliação foram executadas nesta etapa da abordagem de derivação de regras de usabilidade especializadas no contexto multimídia Web.

O agrupamento dos problemas de usabilidade identificados nas interfaces do sistema iClass em aspectos de navegação hipertexto e mídias de informação isoladas realizado nesta etapa é apresentado na Tabela 1.

Tabela 1 – Organização da listagem de problemas de usabilidade identificados nas interfaces do sistema iClass em grupos associados a aspectos de navegação hipertexto e mídias de informação isoladas.

Aspectos de Navegação Hipertexto e Mídias de Informação	**Número de Problemas de Usabilidade Associados**	**Percentagem em Relação ao Total de Problemas de Usabilidade Identificados**
Aspectos de Navegação Hipertexto	37	51%
Texto	16	22%
Gráfico	6	8%
Vídeo	8	11%
Áudio	6	8%
Total	**73**	**100%**

A partir da organização apresentada na Tabela 1 foi possível realizar análises de validação dos problemas de usabilidade identificados. Durante reuniões de análise o experimentador e os avaliadores envolvidos no experimento de Avaliação Heurística reduziram a listagem de problemas gerada inicialmente na etapa Pós-avaliação a um conjunto composto por 65 problemas de usabilidade distintos (Tabela 2).

Tabela 2 – Organização da listagem de problemas de usabilidade validados em função de aspectos de navegação hipertexto e de mídias de informação isoladas.

Aspectos de Navegação Hipertexto e Mídias de Informação	**Número de Problemas de Usabilidade Associados**	**Percentagem em Relação ao Total de Problemas de Usabilidade Identificados**
Aspectos de Navegação Hipertexto	31	48%
Texto	15	23%
Gráfico	6	9%
Vídeo	7	11%
Áudio	6	9%
Total	**65**	**100%**

Além do agrupamento e posterior validação dos problemas de usabilidade identificados, também foi realizado nesta etapa da abordagem de derivação de regras de usabilidade especializadas, o agrupamento das boas soluções de projeto no contexto multimídia Web identificadas nas interfaces do sistema iClass (Tabela 3).

Tabela 3 – Organização das boas soluções de projeto identificadas nas interfaces do sistema iClass em grupos associados a aspectos de navegação hipertexto e mídias de informação isoladas.

Aspectos de Navegação Hipertexto e Mídias de Informação	**Número de Boas Soluções de Projeto Associadas**	**Percentagem em Relação ao Total de Boas Soluções de Projeto Identificadas**
Aspectos de Navegação Hipertexto	2	10,5%
Texto	1	5%
Gráfico	2	10,5%
Vídeo	7	37%
Áudio	7	37%
Total	**19**	**100%**

Análises de validação das soluções de projeto registradas pelos avaliadores do sistema iClass revelaram conformidade integral com as expectativas do experimentador e dos avaliadores envolvidos.

É possível observar a partir da Tabelas 2 que as interfaces do sistema iClass apresentam um menor número de problemas de usabilidade associados as mídias de informação de vídeo e de áudio. Acredita-se que este resultado seja decorrente de uma maior concentração de boas soluções de projeto em aspectos de interface associados a estas mesmas mídias de informação, conforme apresenta a Tabela 3.

4.4. Derivação de Regras de Usabilidade Especializadas no Contexto Multimídia Web

Segundo informação apresentada na introdução da Seção 4, os experimentos de Avaliação Heurística responsáveis pela derivação de regras de usabilidade especializadas no contexto multimídia Web descritos nesta seção envolvem a execução de análises de interfaces de dois sistemas pertencentes ao referido contexto.

Portanto, a etapa de derivação de regras de usabilidade especializadas no contexto multimídia Web referentes a este estudo de caso é apresentada na Subseção 4.8.

4.5. Sistema Jornal da Lílian

O sistema Jornal da Lílian é responsável pela apresentação de informações multimídia geradas em ambientes diversos, como em estúdios de televisão, nas ruas e até mesmo em salas de aula.

O hiperdocumento principal do sistema é apresentado na Figura 4. A porção esquerda do hiperdocumento apresenta uma lista de botões, que quando selecionados, apresentam na porção central deste mesmo hiperdocumento notícias organizadas pelos seguintes assuntos: manchete de capa, últimas notícias, entrevistas, entre outros. A cada reportagem, existem âncoras que apontam para informações de áudio e vídeo associadas àquela matéria em particular. A porção direita apresenta reportagens especiais que são exibidas em um outro hiperdocumento. A porção superior, por sua vez, apresenta informações publicitárias veiculadas no jornal.

Figura 4 – Hiperdocumento principal do sistema Jornal da Lilian.

4.6. Identificação de Problemas de Usabilidade e de Boas Soluções de Projeto

Atividades de Avaliação Heurística sobre um segundo sistema pertencente ao contexto multimídia Web foram executadas neste estudo de caso. Observa-se que a execução de experimentos de Avaliação Heurística sobre um número de sistemas superior a duas unidades proporcione uma amostragem mais significativa dos potenciais problemas de usabilidade pertinentes a um contexto em particular.

Neste sentido, a partir de uma amostragem de problemas de usabilidade de maior representatividade do contexto multimídia Web, torna-se evidente a derivação de um conjunto de regras de usabilidade especializadas e de maior impacto sobre os potenciais problemas de usabilidade cabíveis de existência no referido contexto.

Na primeira etapa do experimento, a etapa Planejamento, foram identificadas as características dos potenciais usuários do sistema Jornal da Lilian: seus conhecimentos sobre o sistema, sobre o contexto de aplicação do sistema (multimídia Web) e sobre aspectos relacionados à computação de forma geral.

Nesta etapa também foram identificadas as regras de usabilidade genéricas de Nielsen consideradas como componentes da usabilidade do sistema (consistência, familiaridade e facilidade de aprendizado).

Por fim, foram selecionados 5 avaliadores para realizar as inspeções nas interfaces do sistema Jornal da Lilian. Os avaliadores selecionados diferiram daqueles envolvidos no experimento de Avaliação Heurística do sistema iClass, porém, a distribuição de conhecimentos em usabilidade equivale entre os dois grupos.

Na etapa Pré-avaliação os avaliadores foram orientados pelo experimentador acerca de aspectos relacionados à execução do experimento, à técnica de Avaliação Heurística e às regras de usabilidade genéricas selecionadas como fundamentais ao sistema Jornal da Lilian.

Na etapa Avaliação os avaliadores executaram inspeções individuais sob as interfaces do Jornal. Como ocorrido no experimento do sistema iClass, os problemas de usabilidade, as regras de usabilidade genéricas associadas e as boas soluções de projeto foram registradas em papel.

Na etapa Pós-avaliação, para concluir o experimento, o experimentador e os avaliadores envolvidos realizaram análises qualitativas e quantitativas sobre os resultados obtidos.

A execução das atividades de inspeção de usabilidade nas interfaces do sistema Jornal da Lilian consumiu aproximadamente um tempo igual a 09 horas e 17 minutos. O número médio de sessões de inspeção de usabilidade utilizadas por cada avaliador foi igual a três e o tempo médio de cada avaliação foi igual a 1 hora e 51 minutos.

Uma listagem final contendo 27 de problemas de usabilidade foi gerada a partir das anotações individuais dos avaliadores, eliminando, assim, eventuais redundâncias existentes. Além disso, uma listagem contendo um total de 24 boas soluções de projeto também foi gerada.

4.7. Análise Qualitativa dos Problemas de Usabilidade e das Boas Soluções de Projeto Identificadas

Análises qualitativas sobre os problemas de usabilidade e as boas soluções de projeto identificadas nas interfaces do sistema Jornal da Lílian revelaram resultados similares àqueles apresentados pelo experimento de Avaliação Heurística do sistema iClass.

Mais especificamente, análises qualitativas dos problemas de usabilidade e das boas soluções de projeto identificadas revelaram uma maior concentração de problemas sobre aspectos de navegação hipertexto e sobre a mídia de informação textual. Além disso, as análises revelaram uma maior concentração de boas soluções de projeto sobre as mídias de vídeo e áudio.

Equivalente ao observado no sistema iClass, as mídias de informação sobre as quais incide uma maior concentração de boas soluções de projeto são aquelas que apresentam uma menor concentração de problemas de usabilidade.

Consideradas essas análises, é possível observar uma carência mais acentuada de regras de usabilidade especializadas no contexto multimídia Web, e mais especificamente, especializadas em aspectos de projeto associados a elementos de navegação hipertexto e à mídia de informação textual.

Esforços de derivação foram, portanto, intensificados sobre as atividades de derivação de regras de usabilidade especializadas em aspectos de navegação hipertexto e na mídia de informação textual.

4.8. Derivação de Regras de Usabilidade Especializadas no Contexto Multimídia Web

Nesta etapa da abordagem de derivação de regras de usabilidade foram redigidas recomendações de projeto especializadas no contexto multimídia Web. O experimentador e os avaliadores envolvidos nos experimentos de avaliação de usabilidade foram responsáveis pela execução das atividades de redação, considerando para isso as listagens de problemas de usabilidade e as listagens de boas soluções de projeto validadas após os experimentos de Avaliação Heurística dos sistemas iClass e Jornal da Lílian.

A listagem final de regras de usabilidade especializadas no contexto multimídia Web foi analisada por pessoas especialistas em comunicação, de forma a então eliminar elementos de redação potencialmente capazes de proporcionar dificuldades ou equívocos de interpretação – principalmente quando utilizadas pessoas não especialistas em usabilidade em atividades de projeto de interfaces que fazem uso dessas regras.

A partir da listagem final de regras de usabilidade especializadas no contexto multimídia Web foi elaborado um experimento de avaliação de usabilidade baseado em checklists compostos pelas regras derivadas.

O objetivo de realizar este experimento de avaliação de usabilidade baseado em checklists foi verificar a adequação das regras de usabilidade derivadas com atividades de identificação de problemas de interação entre usuários e sistemas multimídia Web. Para isso, os resultados obtidos através do uso de checklists e de pessoas não especialistas em usabilidade são comparados quantitativamente com os resultados obtidos através do uso da técnica de Avaliação Heurística, baseada no uso de regras de usabilidade genéricas e de pessoas especialistas em usabilidade.

Com o intuito de tornar os resultados da atividade de análise de adequação das regras de usabilidade derivadas mais significativos foi utilizado nesta etapa um terceiro sistema multimídia Web (*Lecture Browser*).

Vale ressaltar que para os experimentos de análise de adequação das regras de usabilidade ao contexto multimídia Web foram utilizados dois grupos distintos de avaliadores, ambos compostos de cinco pessoas.

O número de problemas de usabilidade identificados no experimento de Avaliação Heurística foi igual a 32 unidades. A análise de validação dos problemas identificados proporcionou a geração de uma listagem final contendo 27 itens associados a aspectos de navegação hipertexto e a mídias de informação isoladas (Tabela 4).

Tabela 4 – Organização por elementos hipertexto e mídias de informação dos problemas de usabilidade identificados em atividades de Avaliação Heurística e de avaliação baseada em checklists nas interfaces do sistema *Lecture Browser*.

Aspectos de Navegação Hipertexto e Mídias de Informação	**Número de Problemas de Usabilidade Associados (Avaliação Heurística | Checklists)**	**Percentagem em Relação ao Total de Problemas de Usabilidade (Avaliação Heurística | Checklists)**
Aspectos de Navegação Hipertexto	13 | 12	49% | 50%
Texto	6 | 5	21% | 20%
Gráfico	2 | 2	08% | 09%
Vídeo	3 | 2	11% | 09%
Áudio	3 | 3	11% | 12%
Total	**27 | 24**	**100% | 100%**

Em contrapartida, o número de problemas de usabilidade identificados no experimento de avaliação de usabilidade baseado em checklists revelou um conjunto de 24 problemas de usabilidade distintos e validados, conforme apresenta a Tabela 4.

A execução do Teste de Wilcoxon revelou que a redução do número de problemas de usabilidade identificados no sistema *Lecture Browser*, consequente da aplicação de *checklists* baseados nas regras de usabilidade derivadas, não é estatisticamente significativa. Em outras palavras, isso significa afirmar que o número de problemas de usabilidade identificados através da execução de um *checklist,* composto por regras de usabilidade derivadas neste experimento e aplicado por pessoas não especialistas em usabilidade, é estatisticamente equivalente ao número de problemas identificados através da execução da técnica de Avaliação Heurística, baseada em regras de usabilidade não especializadas ao contexto e aplicada por avaliadores especialistas em usabilidade.

Portanto, estatisticamente é possível afirmar que as regras de usabilidade derivadas para o contexto multimídia Web são adequadas a atividades de identificação de problemas de usabilidade em interfaces de sistemas pertencentes a este contexto – especificamente quando utilizados avaliadores não especialistas em usabilidade.

5. Conclusões

A utilização de regras de usabilidade genéricas em atividades de projeto de interfaces pode implicar na obtenção de índices de usabilidade não satisfatórios quando comparado aos índices obtidos através do uso de regras de usabilidade especializadas no contexto do sistema avaliado.

Neste contexto, é possível observar uma crescente utilização de atividades componentes de projetos de interfaces que fazem uso de regras de usabilidade especializadas no contexto de aplicação do sistema – principalmente quando não de pessoal especializado em usabilidade para a execução destas atividades.

No entanto, as regras de usabilidade especializadas em contextos de aplicação específicos propostas na literatura são derivadas a partir de mecanismos que contrapõem a necessidade por projetos de baixo custo de execução, e que, por isso, muitas vezes impedem a participação de pessoas especialistas em usabilidade durante a execução de suas atividades.

Uma abordagem de derivação de regras de usabilidade especializadas em contextos de aplicação específicos é proposta neste artigo com o intuito de proporcionar o desenvolvimento de recomendações de projeto de forma rápida e a um baixo custo de execução. Além disso, um estudo de caso com resultados satisfatórios da abordagem proposta é apresentado. Acredita-se que a abordagem proposta contribua significativamente com a execução de atividades de projeto de interfaces que por alguma restrição fazem uso exclusivo de pessoas não especialistas em usabilidade.

Agradecimentos

Os autores agradecem o apoio financeiro recebido do CNPq e da FAPESP.

Referências Bibliográficas

[1] Abowd, G. Classroom 2000: A Experiment with Instrumentation of a Living Educational Environment. *IBM Systems Journal*, 38(4):508-530. 1999.

[2] Abowd, G.; Atkeson, C.; Brotherton, J.; Enqvist, T.; Gully, P.; Lemon, J. Investigating the Capture, Integration, and Access Problem of Ubiquitous Computing in Educational Setting. *In Proceedings of the CHI,* pages 440-447. ACM Press. 1998.

[3] Cybis, W.A.; Tambascia, C.A.; Dyck, A.F.; Villas-Boas, A.L.C.; Oliveira, R.; Freitas, M.; Pagliuso, P.B.B. Abordagem para o Desenvolvimento de Listas de Verificação de Usabilidade Sistemáticas e Produtivas. *Anais do Congresso Latino-Americano de Interação Humano-Computador,* pages 29-40. ACM Press. 2003.

[4] Dias, C. Comparing Usability Evaluation Methods Applied to Corporate Web Portals. *Anais do IHC*, pages 73-83. 2001.

[5] Dix, A.; Finlay, J.; Abowd, G.; Beale, R. Human–Computer Interaction. *Prentice Hall,* Europe, second edition. 1998.

[6] Levi, M.; Conrad, F. A Heuristic Evaluation of a World Wide Web Prototype. *Interactions of ACM*, 3(4):50-61. 1996.

[7] Netto, O.; Pimentel, G. Heurísticas e Guidelines para Apresentação de Hiperdocumentos Multimídia na Web. *Dissertação de Mestrado, Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo*, São Carlos, São Paulo. 2002. Disponível em http://coweb.icmc.usp.br/coweb/mostra.php?ident=5.10

[8] Nielsen, J. The Usability Engineering Life Cycle. *IEEE Computer*, 25(3):12-22. 1992.

[9] Nielsen, J. Usability Engineering. *Academic Press*, London, United Kingdom. 1993.

[10] Nielsen, J. Projetando Web Sites. *Campus*, Rio de Janeiro. 2000.

[11] Nielsen, J; Mack, R. Usability Inspections Methods. *John Wiley & Sons.* 1994.

[12] Pimentel, G.; Abowd, G. Linking by Interaction: a Paradigm for Authoring Hypertext. *In Proceedings of Hypertext,* pages 39-48. ACM Press. 2000.

[13] Pimentel, G.; Netto, O. Experimento de Avaliação Heurística: Estudo de Caso no eClass. *Relatório Técnico, Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo,* São Carlos, São Paulo. Série Computação. 2001. Disponível em http://coweb.icmc.usp.br/coweb/mostra.php?ident=5.10

[14] Rossi, G.; Schwabe, D.; Lyardet, F. User Interface Patterns for Hypermedia Applications. *In Proceedings of AVI,* pages 136-142. ACM Press. 2000.

[15] Sales, M.B.; Cybis, W.A. Desenvolvimento de um Checklist para a Avaliação de Acessibilidade da Web para Usuários Idosos. *Anais do Congresso Latino-Americano de Interação Humano-Computador,* pages 125-133. ACM Press. 2003.

[16] Gibbons, J.D.; Chakraborti, S. Nonparametric Statistical Inference. *Marcel Dekker,* third edition. 1992.

Ranking Global de Páginas Web Basado en Atributos de los Enlaces

Ricardo Baeza-Yates Emilio Davis

Centro de Investigación de la Web
Depto. de Ciencias de la Computación
Universidad de Chile
Blanco Encalada 2120
Santiago 6511224, Chile
E-mail: {rbaeza,edavis}@dcc.uchile.cl

Abstract

Presentamos una variante de Pagerank, el algoritmo más conocido para realizar ranking de páginas Web usando enlaces, que considera distintos atributos de cada enlace para dar distinta importancia a los mismos. Nuestros resultados muestran que la precisión de las respuestas mejora en más de un 10 %.

1. Introdución

Desde que los creadores de Google publicaran el trabajo que menciona por primera vez el algoritmo PageRank [BP98], se ha observado una revolución en el ámbito de la recuperación de información en lo que se refiere a los motores de búsqueda Web. Esta revolución posiblemente ha llevado a la mayoría de los motores a utilizar el algoritmo PageRank o una variación de éste dentro del cálculo de su *ranking*, debido a la efectividad del uso de enlaces.

Por otro lado, han aparecido otros algoritmos que si bien a primera vista no son similares a PageRank, por ejemplo HITS [Kle99], que depende de la consulta del usuario, todos pertenecen a la familia de los rankings normalizados, como se ve en [DHH+02].

Un aspecto que llama la atención, es el hecho que ningún algoritmo del tipo PageRank hace diferencia en los enlaces contenidos en una página determinada. Estos algoritmos "reparten" la misma cantidad de peso en cada enlace, sin tomar en cuenta la relevancia que pueden tener en el resultado de una búsqueda e ignorando los esfuerzos del creador de la página por destacar ciertos enlaces que considere importantes (porque llevan a recursos que él considera más relevantes que otros). Por ejemplo, en la figura 1 las páginas más importantes están en un tono más oscuro y eso muchas veces puede ser inferido a partir de los enlaces.

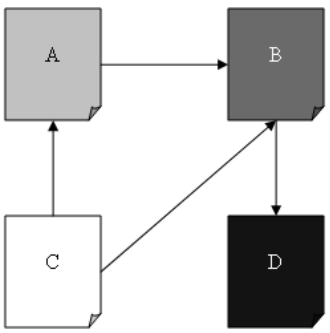


Figura 1: Ejemplo de enlaces entre páginas de distinta importancia.

linkedWebPages.ps

Tomando en cuenta lo explicado anteriormente, se puede concluir que un sistema de *ranking* que pertenezca a la familia de PageRank y que utilice la información que entreguen los generadores de contenido sobre la importancia relativa de los enlaces dentro de una página, superaría la eficacia de PageRank.

En este trabajo presentamos una variante de PageRank que da distinta importancia a los enlaces basados en tres atributos: posición relativa en la página, etiqueta (*tag*) donde se usa el enlace y largo del texto que describe al enlace (*anchor text*). Nuestros resultados muestran que todos estos atributos mejoran el algoritmo original de PageRank, en particular la combinación de ellos.

Comenzamos presentando PageRank, seguido de nuestra variante, WLRank. Luego entregamos la evaluación de la calidad de las respuestas de WLRank y las conclusiones de nuestro trabajo.

2. PageRank

La idea detrás de PageRank es que las *"buenas"* páginas Web son referenciadas mediante enlaces (*links*) por otras páginas, y esto es bastante claro. Si una página contiene información interesante para una persona es problable que en sus páginas la referencie.

El tema de las referencias o citas, no es nuevo, la gran mayoría de publicaciones científicas las utiliza y hay variados enfoques en análisis de citaciones académicas [Gar95] y [Gof71] por ejemplo, y desde un buen tiempo se ha utilizado para jerarquizar tanto las publicaciones como sus autores. Por ejemplo ISI [ISI] o CiteSeer [Cit].

Si bien, tanto páginas Web como publicaciones científicas utilizan un sistema similar para enlazarse, hay una diferencia fundamental. Para que un material académico se publique, éste debe pasar por concienzudos y escrupulosos análisis, mientras, por otro lado, las páginas Web pueden publicarse con una facilidad asombrosa, permitiendo incluso, que programas generen y publiquen grandes volúmenes de páginas con el propósito de aumentar el conteo de referencias para una página en particular. Por esta razón existe *spamming* de enlaces (por ejemplo el fenómeno llamado *Google bombing*) y por ende los algoritmos exactos de uso de enlaces no son públicos.

PageRank [BP98] en su versión simplificada se define así:

$$R(u) = c \sum_{v \epsilon B_u} \frac{R(v)}{N_v}$$

donde u es una página Web, B_u el conjunto de páginas que referencia a u, N_u el número de páginas apuntadas por u y c una constante de normalización. Nótese que el *ranking* de una página es dividido por el número de páginas que apunta para contribuir al *ranking* de éstas. La ecuación es recursiva pero convergente, es decir, si se comienza con un *ranking* arbitrario para cada página y se itera sobre todas las páginas calculando su nuevo *ranking*, al cabo de algunas iteraciones los valores se habrán estabilizado. Más aún, lo que importa es el orden y no los valores exactos, así que basta iterar hasta que no puedan producirse más cambios de orden.

Hay un pequeño problema en esta versión que se ilustra en el siguiente ejemplo: dos páginas se apuntan mutuamente y una tercera apunta a alguna de ellas, entonces durante las iteraciones esas páginas acumularán *ranking* que no distribuirán pues no apuntan a otras páginas. Lo mismo ocurre con autoreferencias o secuencias de enlaces que vuelven a su origen. Para evitar esto se introduce una fuente externa de *ranking* para cada página ($E(u)$) que resuelve el problema, así, PageRank se define como [PBMW98]:

$$R(u) = c \sum_{v \epsilon B_u} \frac{R(v)}{N_v} + E(u)$$

Si vemos PageRank como una navegación aleatoria, donde se escoge un enlace al azar para continuar o se salta a una página al azar de la Web en cualquier momento con probabilidad q, podemos usar $E(u) = q/T$ y $c = 1 - q$, donde T es el número de páginas en la colección. De esta manera la suma de valores de $R(u)$ valdrá siempre 1. Esta es la versión que usamos para explicar nuestra variante.

3. Algoritmo WLRank

WLRank (Weighted Links Rank) asigna el ranking $R(i)$ a la página i de la colección, según las siguientes ecuaciones:

$$R(i) = \frac{q}{T} + (1 - q) \sum_j \frac{W(j,i) R(j)}{\sum_k W(j,k)}$$

$$W(j,i) = L(j,i)(c + T(j,i) + AL(j,i) + RP(j,i))$$

Donde:

- $L(j,i)$ es 1 si existe un enlace desde página j a la página i ($j \in B_i$) y 0 si no existe.

- c es una constante que entrega un peso base a cada enlace.
- $T(j,i)$ es un valor que depende del *tag* en que está inserto el enlace entre la página j y la página i.
- $AL(j,i)$ es el largo del texto del enlace entre la página j y la página i ponderado por una constante d, que es la proporción entre una constante de peso base y una estimación del promedio de los largos de textos de anclaje.
- $RP(j,i)$ es la posición relativa del enlace dentro de la página ponderada por una constante b.

La figura 2 muestra un ejemplo concreto de la importancia que tendrían distintos enlaces de acuerdo a los atributos definidos.

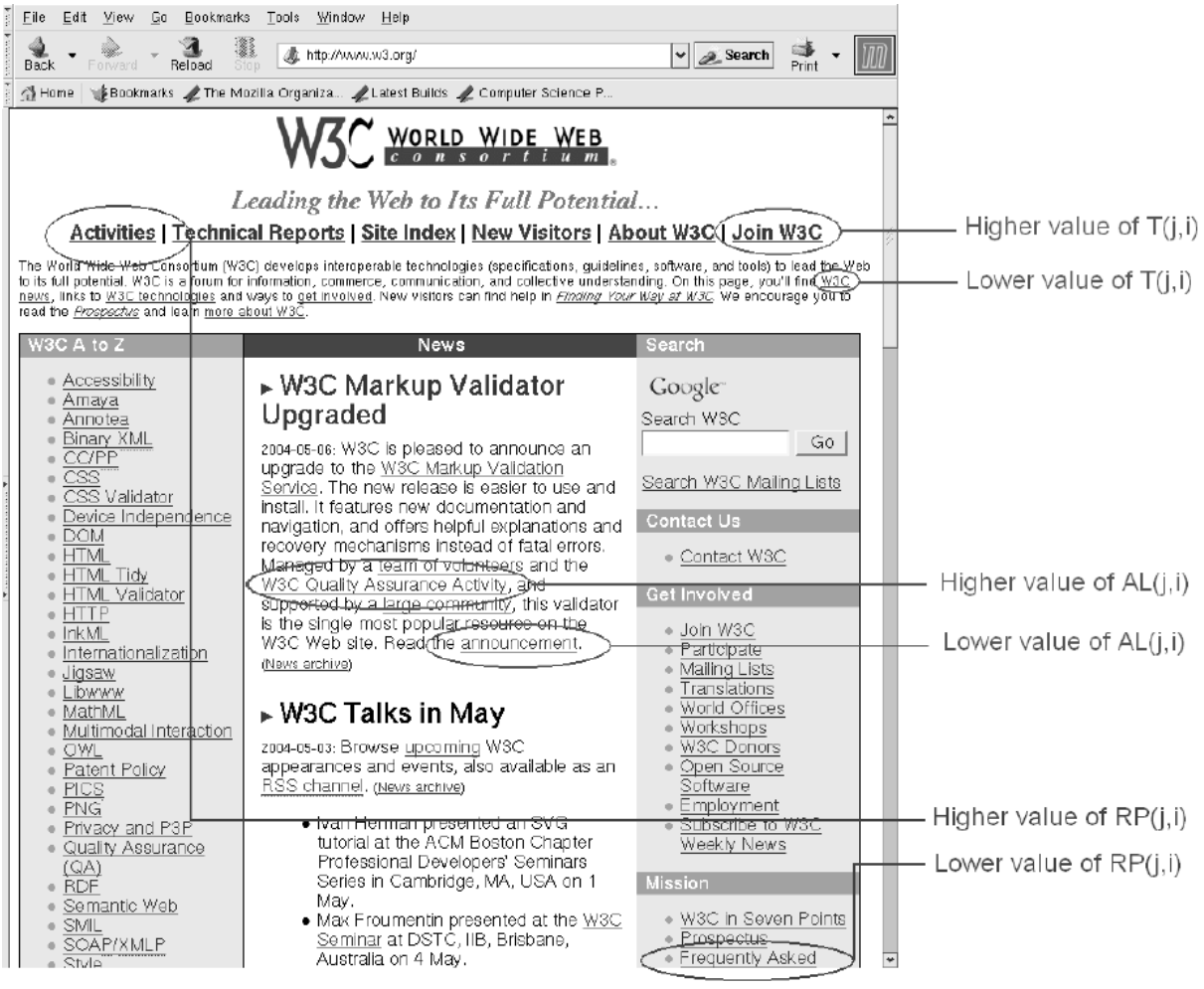


Figura 2: Ejemplo de distintos atributos de enlaces y su importancia relativa.

El *ranking* $R(i)$ corresponde a la probabilidad de un usuario de llegar a la página i, y se compone de dos términos. El primero corresponde al "salto" aleatorio desde cualquier página a ésta (distribución uniforme con probabilidad q/T) y el segundo, al peso ponderado con que aporta cada página que apunta a i.

Como se ve, si el término $W(j,i)$ fuese igual a $L(j,i)$ se obtiene PageRank. A continuación se explican las modificaciones.

El término $T(j,i)$ será una serie de constantes dependiendo del *tag* en que se encuentre el enlace, por ejemplo, un enlace que se encuentre entre *tags* $< h1 >$, tendrá un valor $T(j,i)$ alto, un poco más bajo si está entre *tags* $< h2 >$, etc. Lo mismo para otros *tags* de énfasis como $< strong >$, $< b >$, etc.

El término $AL(j,i)$ dará más valor a los enlaces en que el creador de la página haya explicado con más detalle el recurso que está siendo apuntado.

Por último, el término $RP(j,i)$ valorará más los enlaces que se ubiquen más hacia el principio de la página que hacia el final.

El algoritmo para calcular WLRank es el siguiente, basado en el de PageRank:

- Inicializar WLRank para cada página en $1/TotalPaginas$.
- Calcular la suma de los pesos de los enlaces salientes para cada documento.
- Iterar N veces (con N suficientemente grande para asegurar convergencia).
  - Para cada página i en la colección
    - Para cada página j que es apuntada por i
      - `WLRank(j) += WLRank(i) * PesoEnlace(i,j) / SumaPesosEnlacesDesde(i)`
  - Para cada página i en la colección
    - `WLRank(i) = q/TotalPaginas + (1-q)WLRank(i)`

Según [PBMW98] $N = 50$ asegura convergencia para colecciones de 300 millones de enlaces. Aunque WLRank es más complicado, en nuestros experimentos hemos necesitado valores menores de N para obtener convergencia.

4. Evaluación

En el área de recuperación de la información, existen dos parámetros importantes para evaluar la calidad de las respuestas de un motor de búsqueda.

- Recuperación (*recall*): Razón entre el número de documentos relevantes recuperados y el número total de documentos relevantes. Para poder calcular este indicador es necesario contar con una colección de documentos, un conjunto de consultas y las respuestas relevantes para cada consulta, lo que en la Web es difícil de generar.
- Precisión: razón entre el número de documentos relevantes recuperados y el número total de documentos recuperados. A diferencia del indicador anterior, para calcular éste sólo se necesita una colección de documentos y usuarios con la capacidad de realizar consultas y determinar si un resultado es relevante o no, dentro de un cierto número de respuestas.

Para probar WLRank se realizó una colecta sobre la Web chilena y usuarios de prueba que hicieron consultas al sistema y evaluaron si las respuestas que entrega son pertinentes o no, dentro de la primera página de resultados (cada página contiene diez respuestas). Cada persona evaluaba los resultados sin conocer cual caso era el que correspondía en cada conjunto de resultados. Se buscó un conjunto de usuarios de prueba de dominios de conocimiento variados de manera de simular un universo representativo de los usuarios de la Web.

La colecta sobre la Web chilena utiliza como punto de partida el conjunto de URLs que mantiene NIC Chile[1]. Nuestro sistema de búsqueda fue alimentado con este conjunto de URLs y se realizó el ciclo de recolección hasta contar con un conjunto de 460 mil páginas indexadas. Este índice se utilizó para realizar todas las pruebas.

En esta colecta se calculó WLRank usando peso unitario para el peso base ($c = 1$), el factor de posición relativa ($b = 1$), el peso de los tags <b> y <h1> (sin considerar otros tags), y el factor del largo del texto de anclaje considera largo promedio del texto de anclaje de 100 caracteres ($d = 1/100$). Además se calculó WLRank usando cada una de las modificaciones a PageRank, es decir, un WLRank agregando sólo la posición relativa, otro sólo agregando los *tags* y finalmente uno utilizando sólo el largo del texto de anclaje, usando todos ellos sobre la misma colección de páginas para que la comparación sea válida.

A cada usuario se le asignó como tarea realizar tres consultas de su elección al sistema (se supone que será un usuario experto para cada uno de esas consultas), utilizando PageRank y WLRank (en sus cuatro formas) como sistema de *ranking*. El usuario debió determinar, dentro de la primera página de resultados, cuáles respuestas fueron pertinentes a la consulta que realizó. En el cuadro 1 se observan los resultados de esta prueba utilizando PageRank y WLRank completo. En los cuadros 3 y 4 del apéndice se muestran los resultados utilizando los WLRank parciales.

Con estos datos se calculó la precisión de PageRank y WLRank (completo y parciales), los resultados se observan en la figura 3.

Para determinar si WLRank es una mejora a PageRank se pueden comparar ambos con un *ranking* "perfecto". Definimos como un *ranking* "perfecto" a un sistema de ordenamiento que logre ubicar dentro de las primeras páginas de resultados sólo respuestas relevantes. Con esto vemos, en el cuadro 2, el desempeño de PageRank y WLRank contra un *ranking* "perfecto". Como se observa, WLRank mejora en aproximadamente un 12 % a PageRank.

[1] Entidad encargada de la administración del dominio .cl, http://www.nic.cl

Consulta	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**R8**	**R9**	**R10**
aspirina	×	√	×	×	√	√	×	×	√	×
educacion	√	√	×	×	×	×	×	×	×	×
bicicleta	√	√	√	×	×	√	×	×	×	√
computacion	√	√	√	√	√	√	√	×	×	√
mascota	×	×	×	×	×	√	√	√	√	√
starwars	√	×	√	√	√	√	√	√	√	√
todocl	√	×	×	×	×	√	×	√	×	×
cosmeticos	×	√	×	×	×	√	√	×	√	×
hulk	√	√	√	√	√	√	√	√	√	√
quake	√	√	×	×	×	×	×	×	×	×
mozilla	×	×	×	×	×	×	×	×	×	×
cine	×	√	×	√	×	×	×	√	√	×
"andres bello"	×	√	√	×	×	×	×	×	×	√
citibank	×	√	×	×	×	×	×	×	×	×
musica	√	√	×	√	√	√	√	√	√	√
futbol	×	√	√	√	√	√	√	√	√	√
caballo corralero	×	×	√	×	√	√	√	×	√	√
biotecnologia	√	×	×	√	√	√	√	√	√	√
cuadernos	√	√	×	×	×	×	×	√	×	√
viajes	×	×	√	√	√	√	√	√	√	×
aspirina	√	×	×	×	√	×	×	√	×	×
educacion	√	×	√	×	×	×	×	×	×	√
bicicleta	√	√	√	×	×	√	√	×	×	×
computacion	√	√	√	√	√	√	√	√	√	×
mascota	×	×	×	×	×	√	√	√	√	√
starwars	√	×	√	√	√	√	√	√	√	√
todocl	√	×	×	×	×	√	√	×	×	×
cosmeticos	×	√	√	√	×	√	√	√	×	×
hulk	√	√	√	√	√	√	√	√	√	√
quake	√	√	×	×	×	×	×	×	×	√
mozilla	×	×	×	×	×	×	×	×	×	×
cine	×	√	√	√	√	×	×	√	×	√
"andres bello"	×	√	√	√	×	×	×	×	×	×
citibank	×	√	×	×	×	×	×	×	×	×
musica	√	√	√	√	√	√	√	√	√	√
futbol	√	×	√	√	√	√	√	√	√	√
caballo corralero	×	×	×	√	√	√	√	×	√	√
biotecnologia	×	√	√	√	×	√	√	√	√	√
cuadernos	√	√	×	×	×	×	√	×	√	√
viajes	√	×	√	√	√	√	√	√	√	√

Cuadro 1: Resultados de las pruebas usando PageRank (arriba) y WLRank completo (abajo).

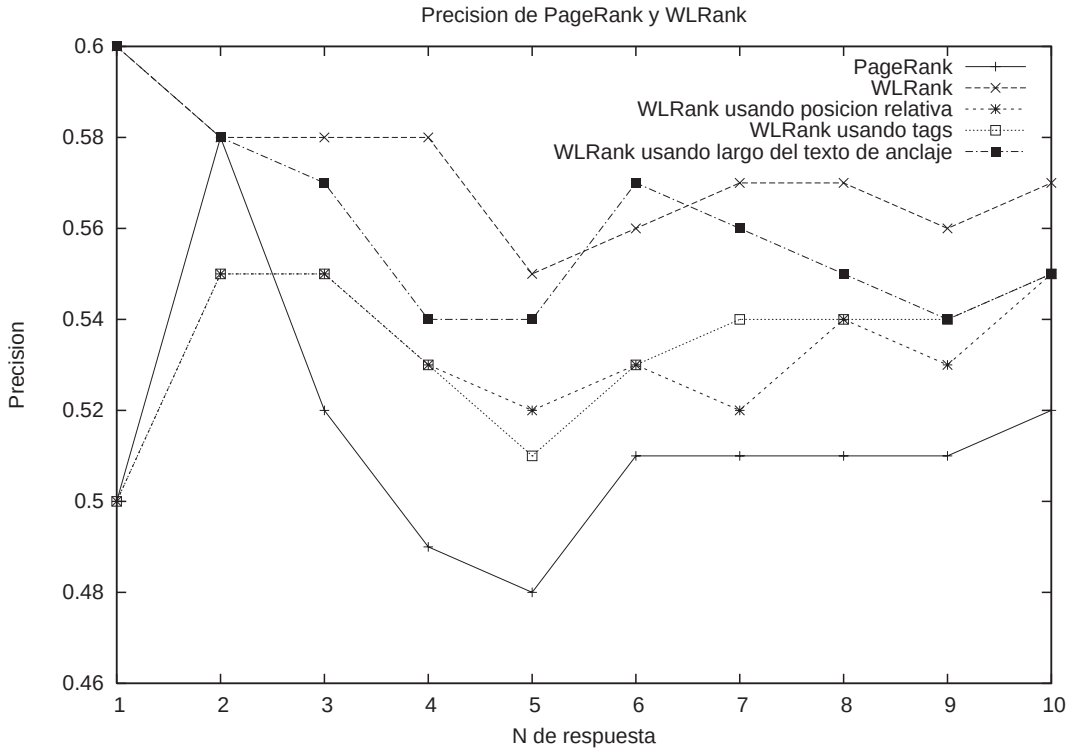


Figura 3: Precisión de PageRank y WLRank.

5. Conclusiones

Nuestros resultados muestran que el uso de enlaces de peso variable puede mejorar la precisión en más de un 10 %. Esto indica que usar estos atributos u otros similares puede mejorar el ranking. En nuestro caso la posición relativa no fue tan efectiva, lo que puede indicar que dado el diseño actual de páginas Web, un enlace puede estar lógicamente arriba pero no al comienzo del HTML asociado.

Trabajo futuro incluye una evaluación con más usuarios y un análisis detallado de los factores de importancia de cada atributo, lo que permitirá mejorar aún más WLRank. Cabe hacer notar que los valores dependen de los creadores de páginas Web y no de los que navegan por aquellas páginas. Por otro lado, estos factores deben ser confidenciales y posiblemente modificados permanentemente para evitar enlaces que intentan engañar al buscador (*spamming* de enlaces).

Respuesta	**Perfecto**	**Perfecto - Pagerank**	**Perfecto - WLRank**	**Mejora (%)**
1	1	0.5	0.4	-25
2	1	0.43	0.43	0
3	1	0.48	0.42	9
4	1	0.51	0.43	8
5	1	0.52	0.45	9
6	1	0.49	0.44	10
7	1	0.49	0.43	12
8	1	0.49	0.43	12
9	1	0.49	0.44	11
10	1	0.49	0.44	11
Error total	**0**	**4.89**	**4.29**	12

Cuadro 2: Comparación de PageRank y WLRank contra un ranking "perfecto".

Referencias

[BP98] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. In *7th WWW Conference*, Brisbane, Australia, April 1998.

[Cit] CiteSeer. http://citeseer.com.

[DHH+02] Chris Ding, Xiaofeng He, Parry Husbands, Hongyuan Zha, and Horst Simon. Pagerank, hits and a unified framework for link analysis. *LBNL Tech Report 49372*, 2001-2002.

[Gar95] Eugene Garfield. New international professional society signals the maturing of scientometrics and informetrics. *The Scientist*, 9(16), 1995.

[Gof71] William Goffman. A mathematical method for analyzing the growth of the scientific discipline. *Journal of the ACM*, 18(2):173–185, 1971.

[ISI] ISI. Citation index, http://isinet.com.

[Kle99] J. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 48:604–632, 1999.

[PBMW98] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

Apéndice: Resultados de las Pruebas Parciales

Consulta	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**R8**	**R9**	**R10**
aspirina	×	√	×	×	√	×	×	√	×	×
educacion	√	×	√	×	×	×	×	×	×	×
bicicleta	√	√	√	×	×	√	×	×	×	√
computacion	√	√	√	√	√	√	√	√	×	√
mascota	×	×	×	×	×	√	√	√	√	√
starwars	√	×	√	√	√	√	√	√	√	√
todocl	√	×	×	×	×	×	×	×	×	×
cosmeticos	×	√	√	√	×	√	√	√	×	√
hulk	√	√	√	√	√	√	√	√	√	√
quake	√	√	×	×	×	×	×	×	×	×
mozilla	×	×	×	×	×	×	×	×	×	×
cine	×	√	×	√	√	×	×	√	√	√
"andres bello"	×	√	√	×	×	×	×	√	×	×
citibank	×	√	×	×	×	×	×	×	×	×
musica	√	√	√	√	√	√	√	√	√	√
futbol	√	×	√	√	√	√	√	√	√	√
caballo corralero	×	×	×	√	√	√	√	×	√	√
biotecnologia	×	√	√	×	√	√	√	√	√	√
cuadernos	√	√	×	×	×	×	×	√	√	√
viajes	×	×	√	√	√	√	√	√	√	√

Cuadro 3: Resultados de las pruebas usando WLRank con posición relativa.

Consulta	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10
aspirina	×	√	×	×	√	√	×	×	√	×
educacion	√	×	√	×	×	×	×	×	×	×
bicicleta	√	√	√	×	×	×	√	×	×	×
computacion	√	√	√	√	√	√	√	×	×	√
mascota	×	×	×	×	×	√	√	√	√	√
starwars	√	×	√	√	√	√	√	√	√	√
todocl	√	×	×	×	×	√	√	×	×	×
cosmeticos	×	√	√	×	×	×	√	√	√	√
hulk	√	√	√	√	√	√	√	√	√	√
quake	√	√	×	×	×	×	×	×	×	×
mozilla	×	×	×	×	×	×	×	×	×	×
cine	×	√	√	√	×	√	×	√	√	√
"andres bello"	×	√	√	×	×	×	×	×	×	×
citibank	×	√	×	×	×	×	√	×	×	×
musica	√	√	√	√	√	√	√	√	√	√
futbol	×	√	√	√	√	√	√	√	√	√
caballo corralero	×	×	×	√	√	√	√	×	√	√
biotecnologia	√	×	×	√	√	√	√	√	√	√
cuadernos	√	√	×	×	×	×	×	√	×	√
viajes	×	×	√	√	√	√	√	√	√	√
aspirina	√	×	×	×	√	√	×	×	×	√
educacion	√	×	√	×	×	×	×	×	√	×
bicicleta	√	√	×	√	×	√	×	√	×	√
computacion	√	√	√	√	√	√	√	×	×	√
mascota	×	×	×	×	×	√	√	√	√	√
starwars	√	×	√	√	√	√	√	√	√	√
todocl	√	×	×	×	×	√	×	×	×	×
cosmeticos	√	√	√	×	×	√	√	√	×	√
hulk	√	√	√	√	√	√	√	√	√	√
quake	√	√	×	×	×	×	×	×	×	×
mozilla	×	×	×	×	×	×	×	×	×	×
cine	×	√	√	×	√	√	×	√	×	×
"andres bello"	×	√	√	×	×	×	×	×	×	×
citibank	×	√	×	√	×	×	×	×	×	×
musica	√	√	√	√	√	√	√	√	√	√
futbol	×	√	√	√	√	√	√	√	√	√
caballo corralero	×	×	×	√	√	√	√	×	√	√
biotecnologia	√	×	√	×	√	√	√	√	√	√
cuadernos	√	√	×	×	√	×	×	×	×	√
viajes	×	×	√	√	√	√	√	√	√	√

Cuadro 4: Resultados de las pruebas usando WLRank con *tags* (arriba) y con largo del texto de anclaje (abajo).

Modelagem Adaptativa de Aplicações Complexas

Almir Rogério Camolesi [1,2], João José Neto [1]

[(1)] Departamento de Engenharia de Computação e Sistemas Digitais (PCS),
Escola Politécnica (POLI), Universidade de São Paulo (USP),
Av. Luciano Gualberto, Trav. 3, nº 380 - 05.508-900 - São Paulo - SP – Brasil

[(2)] Coordenadoria de Informática, Instituto Municipal de Ensino Superior de Assis (IMESA),
Fundação Educacional do Município de Assis (FEMA)
Av. Getulio Vargas, 1200 – 19.807-634 – Assis – SP - Brasil

almir.camolesi, joao.jose@poli.usp.br

Abstract

This paper presents the ISDL-Adp model and its adaptive actions. This model is the result of extending the concepts of non-adaptive ISDL devices by using the concepts of adaptive devices. This paper shows the basic structure of adaptive actions and its use in modeling of behavior of distributed adaptive system.
Key words: non adaptive devices, adaptive devices, ISDL, ISDL-Adp, system modeling.

Resumo

Este artigo apresenta a estrutura do modelo ISDL-Adp e de suas ações adaptativas. Tal modelo é fruto da extensão do modelo não-adaptativo ISDL aos conceitos de mecanismos adaptativos. Neste trabalho serão apresentadas a estrutura básica das funções adaptativas e a sua utilização na modelagem de sistemas distribuídos adaptativos.
Palavras-chave: dispositivos não-adaptativos, dispositivos adaptativos, ISDL, ISDL-Adp, modelagem de sistemas.

1. Introdução

Ao desenvolver uma aplicação os projetistas não desejam construir sistemas complexos, mas devido ao grande número de funcionalidades que um sistema deve desempenhar e as necessidades requeridas pelos usuários a maioria das aplicações iniciam com um ciclo de vida contendo um mínimo de funcionalidades requeridas para execução. Entretanto muitos softwares estão acoplados a ambientes complexos e, mais importante ainda, à ambientes que incluem seres humanos. A introdução de novos sistemas de software e as mudanças no ambiente de operação criando mudanças recíprocas nas interações do sistema tornam o desenvolvimento das aplicações cada dia mais complexo. Todos estes fatores motivam os projetistas a buscarem técnicas que facilitem o projeto de tais aplicações.
Muito trabalho vem sendo realizado nos últimos anos em buscas de técnicas e ferramentas para o auxilio aos projetistas no desenvolvimento de suas aplicações. Dentre os modelos empregados pode-se citar metodologias e técnicas como *UML* [1], *Statecharts* [2] e Técnicas de Descrição Formal (*TDFs*) como *LOTOS* [3], E*stelle* [4], *SDL* [5], etc..
Na década de 90 um grupo de pesquisadores iniciou um estudo buscando algumas deficiências em relação a aplicações de TDF no projeto de sistemas, gerando ao mesmo tempo um conhecimento mais profundo das necessidades de modelagem de sistemas distribuídos. Fundamentado nos estudos realizados desenvolveram um conjunto de conceitos arquitetônicos [6] que formaram a base conceitual do modelo *Interaction System Design Language (ISDL)* [7] para suportar o projeto de sistemas telemáticos, como, por exemplo, serviços e protocolos do modelo *OSI*. Atualmente este modelo é utilizado em diversas áreas de conhecimento, como sistemas distribuídos, telemáticos e reengenharia de processos de negócios.
Outros trabalhos focados na especificação de aplicações que tem o seu comportamento modificado durante o tempo de execução foram desenvolvidos. Em [8] é apresentado um mecanismo fundamentado nos conceitos básicos de Autômatos Finitos. Em [9] foi desenvolvida uma ferramenta para especificação de aplicações dinâmicas utilizando a técnica de *Statechart* como base. Atualmente pode-se citar trabalhos como [10] e [11], que buscam o desenvolvimento de técnicas e metodologias para o desenvolvimento de aplicações dinâmicas e reconfiguráveis.
Neste contexto é apresentado o modelo *Adaptive Interaction System Design Language (ISDL-Adp)* que tem por objetivo a extensão do Modelo ISDL aos conceitos de mecanismos adaptativos visando a especificação do comportamento de sistemas distribuídos complexos que possuem características reconfiguráveis presentes em seu comportamento. Tal extensão fundamenta-se na estrutura básica do Modelo ISDL [7] estendida aos conceitos de Mecanismos Adaptativos proposto em [12]. Este artigo encontra-se estruturado da seguinte forma: a seção 2 ilustra os conceitos básicos do Modelo ISDL; na seção 3 é descrita a estrutura do modelo ISDL-Adp e de suas

funções adaptativas, na seção 4 é realizado um exemplo ilustrativo e, por fim, na seção 5, são tecidas algumas conclusões e trabalhos futuros.

2. Modelo não adaptativo ISDL

O principal aspecto a ser considerado no modelo formal desenvolvido é a possibilidade de representação das possíveis mudanças que ocorrem em um determinado comportamento durante o tempo de execução. Tais mudanças podem ocorrer no conjunto de atividades desempenhado por uma aplicação, no conjunto de dados manipulados, no tempo em que uma determinada atividade é executada ou na localização de ocorrência da mesma. Como modelo subjacente ao mecanismo proposto foi utilizado o *Interaction System Design Language* (ISDL).

ISDL é um modelo genérico que foi desenvolvido para modelagem de diversos tipos de sistemas distribuídos, como processos de negócios, aplicações telemáticas e redes de comunicação. Tal modelo fundamenta-se na busca de minimização de limitações existentes em métodos formais [13]. A modelagem de um sistema em ISDL consiste basicamente de dois modelos: um modelo de entidades e um modelo de comportamentos.

O modelo de entidade representa quais partes do sistema que são consideradas, e como elas são interconectadas. Dois conceitos são usados em um modelo de entidade: entidade e pontos de interação. Uma entidade representa um sistema (parte) que desempenha alguma função ou comportamento, como, por exemplo, um componente de software ou departamento de uma empresa. Um ponto de interação representa um mecanismo que possibilita que uma entidade possa interagir com outras entidades, como, por exemplo, um cliente de correio eletrônico.

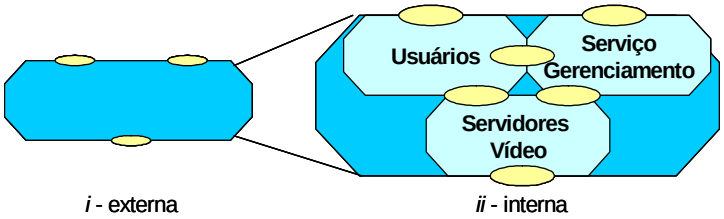


Figura 1 - Exemplo de Modelo de Entidade.

De uma perspectiva externa, um sistema é modelado por uma simples entidade, tendo um ou mais pontos de interação. A Figura 1*i*, por exemplo, representa um modelo de entidade de sistema de vídeo sob demanda (VoD) [17] onde as interações ocorrem com o sistema por meio de três pontos de interação. Estes pontos de interação poderiam, por exemplo, representar, as requisições de serviços realizadas por um usuário, as interações realizadas pelo administrador do sistema e os serviços de manutenção de mídias. Uma entidade é graficamente representada por um retângulo com os cantos entrecortados. Um ponto de interação é graficamente representado por uma elipse sobreposta às entidades que compartilham (eg., via de comunicação) os pontos de interação.

De uma perspectiva interna, um sistema é modelado com uma composição de partes funcionais. Por exemplo, estas partes podem representar sub componentes ou departamentos de uma empresa. A perspectiva interna do VoD é apresentada na Figura 1*ii*. Esta figura ilustra o VoD dividido em três partes: um serviço genérico do usuário, um provedor que notifica os usuários sobre os serviços que estão disponíveis: autenticação, mídias disponíveis, etc.., e um serviço provedor de mídias que são requisitadas pelos usuários.

O modelo de comportamento representa o comportamento, ou funcionalidade, de cada entidade em um correspondente modelo de entidade. Três conceitos são usados: ação, interação e relações de causalidade. Uma ação representa alguma atividade executada por uma entidade simples. Considere por exemplo, a ação do serviço provedor de mídias que é responsável pela entrega das mídias aos usuários do serviço. Uma interação representa uma atividade executada por duas (ou mais) entidades. Uma contribuição de interação representa a participação de uma entidade individual em uma atividade comum. Considere, por exemplo, a interação entre o serviço de requisição de usuários e a apresentação de uma lista de filmes (mídias) disponíveis para apresentação.

Fundamentado nos conceitos apresentados em [12] o conjunto de conceitos arquitetônicos do modelo ISDL constituem um mecanismo não-adaptativo dirigido por regras cujo comportamento depende exclusivamente de um conjunto finito de regras que determinam, para cada possível configuração corrente do mecanismo, a sua próxima configuração. Desta forma, o mecanismo não-adaptativo ISDL é descrito como sendo uma sêxtupla, e é representado formalmente por ISDL = (Ac, RC, Σ, c_0, A, NA), onde:

- ISDL é um mecanismo não-adaptativo dirigido por regras cuja operação define um comportamento constituído de um conjunto de Relações de Causalidades RC;
- Ac é o conjunto de todas as possíveis ocorrências de ações, e $c_0 \in$ Ac determina o conjunto de ações do comportamento inicial. Os possíveis comportamentos alcançados são representados por intermédio da conjunção cruzada de suas execuções parciais. A conjunção cruzada e as execuções parciais são denotadas formalmente por ⊗ e $e\chi$. Vamos supor a conjunção cruzada de dois comportamentos parciais *EE1* e *EE2* que constituem o comportamento *EE*. *EE* consiste de todas as possíveis execuções $e\chi$, sendo que $e\chi$ é a conjunção de duas execuções compatíveis $e\chi 1$ e $e\chi 2$, com $e\chi 1 \in EE1$ e $e\chi 2 \in EE2$. Conseqüentemente, o comportamento *EE* consiste de todas as possíveis construções alternativas que satisfaçam ao mesmo tempo

uma (sub-) construção de *EE1* e uma (sub-) construção de *EE2*.

- ε denota a cadeia vazia, que representa o elemento neutro do conjunto a que pertence, em relação à operação de concatenação;
- Σ é o conjunto finito de todos os possíveis eventos que formam as cadeias de entrada válidas para ISDL, com $\varepsilon \in \Sigma$;
- $A \subseteq Ac$ é o conjunto de ocorrências de ações;
- $F = Ac - A$ é o subconjunto das não ocorrências de ações;
- $w = w_1 ... w_n$ é uma cadeia de entrada, onde $w_k \in \Sigma-\{\varepsilon\}$, k = 1,..., n com $n \geq 0$;
- NA é um conjunto finito (com $\varepsilon \in NA$) de todos os possíveis símbolos gerados como saída do mecanismo ISDL. Na prática, os símbolos de saída contidos em NA podem ser obtidos por meio de chamadas de procedimento, assim uma saída gerada por uma aplicação de qualquer relação de causalidade básica pode ser interpretada como chamada do procedimento correspondente;
- RC é um comportamento definido por um conjunto de ações e interações de comportamentos e suas relações de causalidades. Um comportamento representa um conjunto de atividades que a entidade (conceito abstrato que modela um sistema, ou parte de um sistema) pode executar. O conceito de ação foi introduzido para representar uma atividade executada por um único sistema em um dado nível de abstração.

Atributos de informação, tempo e localização podem ser adicionados a uma (inter)ação para modelar, respectivamente, os resultados estabelecidos por alguma atividade, num determinado instante de tempo, em uma determinada localização. A ocorrência de uma (inter)ação representa o término com sucesso de uma atividade. Uma inter(ação) é atômica desde que a mesma ocorra, algum resultado é estabelecido e torna-se disponível em um determinado tempo numa determinada localidade para todas as entidades envolvidas na atividade. Caso contrário, nenhum resultado é estabelecido e nenhuma atividade pode se referir a qualquer resultado intermediário de uma atividade.

Uma ação é graficamente ilustrada por um círculo (ou elipse). Uma contribuição de interação é graficamente representada por um segmento de círculo (ou elipse), que reflete que múltiplas entidades contribuem para a interação. Os atributo de informação (ι), de tempo (τ) e de localização (λ) são representados dentro de caixas-texto anexadas as (inter)ações. Construções podem ser definidas nas possíveis saídas dos valores de ι, τ e λ . No caso de uma interação, cada contribuição de interação define a construção da correspondente entidade, cada uma com os respectivos valores de ι, τ e λ devem satisfazer todas as construções das entidade envolvidas, caso contrário a interação não pode ocorrer. Também são permitidos múltiplos valores para algum atributo, e uma escolha não determinística entre estes valores é assumida.

Desta forma, o conjunto de ações e interações que definem um comportamento ISDL é dado por uma relação de causalidade:

$$\langle\langle a, \mathrm{I}, \mathrm{T}, \Lambda, \varsigma\rangle, \Gamma, \upsilon, \langle \textit{I-Refs, T-Refs, L-Refs, ITLRefs}\ \rangle, \langle \textit{I-Caus, T-Caus, L-Caus, ITL-Caus}\rangle\rangle$$

onde:

- $a \in \boldsymbol{A}$ é o nome da ação, que a identifica unicamente no sistema, e faz parte de A (conjunto de ocorrências de ações);
- I, T , Λ são, respectivamente, os valores dos domínios de informação, tempo e localização da ação *a*;
- $\varsigma \subseteq \mathrm{I} \times \mathrm{T} \times \Lambda$ é a combinação dos valores dos domínios da ação *a*;
- $\Gamma \in \boldsymbol{CC}$ é a condição de causalidade de *a* e faz parte de ***CC*** (conjunto de condições de causalidades disjuntivas);
- *I-Refs*, *T-Refs*, *L-Refs*, *ITLRefs* são, respectivamente, os conjuntos dos atributos de informação, tempo, localização e de combinação destes atributos;
- I-Caus, T-Caus, L-Caus, ITL-Caus são, respectivamente, os conjuntos de relações de causalidade de informação, de tempo, de localização e de combinações destes atributos;

Uma relação de causalidade é associada com cada (inter)ação, modelando a condição para esta inter(ação) ocorrer. Três condições básicas para a ocorrência de uma ação *a* são identificadas: $b \rightarrow a$; a ação *b* deve ocorrer antes da ação *a*; $\neg b \rightarrow a$; a ação *b* não deve ocorrer antes, nem simultaneamente com a ação *a*; e $\surd \rightarrow a$; a ação *a* é habilitada a ocorrer.

O operador *e* (∧) e o operador *ou* (∨) podem ser usados para modelar condições de causalidades complexas. Por exemplo, $b \vee \neg c \rightarrow a$ representa que a ação *a* ocorre depois da ação *b* ter ocorrido e a ação *c* ainda não ter ocorrido. Além disso, um atributo de probabilidade para cada condição pode ser usado para representar a probabilidade da inter(ação) acontecer quando a condição é satisfeita.

O conceito de relação de causalidade permite a modelagem de muitas diferentes relações entre ações. Isto é conveniente para representar estes relacionamentos diretamente, ao invés da composição de relações de causalidade de ações individuais. A Figura 2 ilustra graficamente algumas relações comuns entre duas ou três ações. Os

operadores ∧ e ∨ são graficamente representados, respectivamente, pelos símbolos ■ e □ . A condição inicial √ é representada por uma seta com nenhuma ação anterior ligada a mesma.

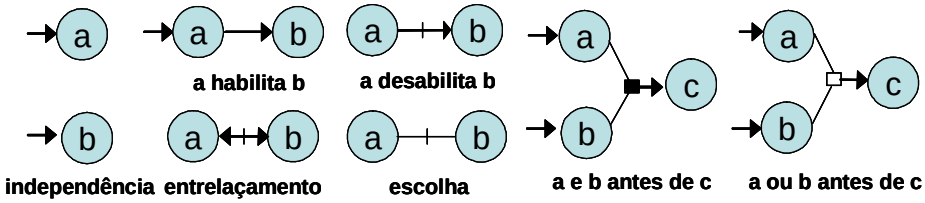


Figura 2 - Algumas ações e relações de causalidade.

O modelo ISDL suporta duas técnicas ortogonais para estruturação de comportamentos por meio da composição de subcomportamentos menores e mais simples: estrutura orientada a causalidade e estrutura orientada a restrição. A estrutura de comportamentos orientada a causalidade é fundamentada na decomposição de uma relação de causalidade em uma construção sintática, que permite definir uma ação e suas relações de causalidades em diferentes subcomportamentos.

A estrutura de comportamentos orientada a restrição é fundamentada na decomposição de uma ação em uma ou mais interações, que permite que uma ação seja definida para um comportamento como uma composição de interações de sub-compor-tamentos. Esta técnica pode ser usada para decompor condições e construções complexas na execução de uma ação em simples sub-condições e sub-construções que são assumidas para contribuições de interações definidas em sub-comportamentos distintos. Além disso, a estrutura orientada a restrição é necessária para estruturar um comportamento em sub-comportamentos de forma que o mesmo possa ser atribuído a diferentes entidades, desde que as entidades possam se comunicar por meio de interações.

A Figura 3 mostra a modelagem do acesso, da requisição de serviços e do gerenciamento de requisições em que ambas às técnicas de estruturação de comportamento são usadas. Por meio da estrutura orientada à restrição é modelado o acesso do usuário ao sistema, o envio e recebimento de requisições e a execução dos serviços solicitados. Na referida modelagem o usuário realiza a solicitação de acesso ao sistema por meio da interação *Login.* O sistema após receber e validar a requisição de acesso habilita o usuário a fazer requisições ao subcomportamento *Gerenciamento Serviços* que os disponibiliza aos usuários (ex: Fim Serviço). Utilizando-se da estruturação de comportamento orientada a causalidade foi modelada as interações existentes entre os *Usuários* e os subcomportamentos *Serviço Autenticação* e *Serviço Gerenciamento*. Enquanto que a estruturação orientada a causalidade foi utilizada para representar a habilitação da interação *Requisição* do subcomportamento *Serviço Gerenciamento* por meio de pontos de entrada e saída.

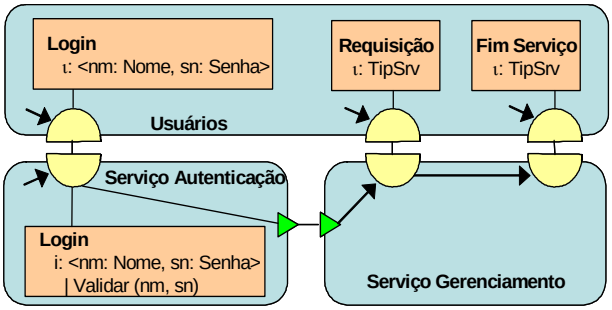


Figura 3 - Exemplo de Estruturação de Comportamento.

Em [14] uma ferramenta gráfica para AMBER, um dialeto de ISDL constituído para reengenharia de processos, foi desenvolvido e o Projeto Friends [15] tem adaptado e estendido esta ferramenta para suportar a modelagem de componentes de software e suas composições para sistemas telemáticos.

3. Modelo Adaptativo ISDL-Adp

A arquitetura básica do modelo *Adaptive Insteraction System Design Language* (ISDL-Adp) [16] representada na Figura 4 constitui-se de um núcleo não-adaptativo ISDL [7], envolvido por uma camada de ações adaptativas que tem por finalidade a realização de mudanças das características do sistema representado. A camada adaptativa é formada por um conjunto de ações adaptativas anteriores e posteriores semelhantes as ações definidas para os Autômatos Adaptativos [8]. As ações adaptativas têm por objetivo realizar mudanças no comportamento da especificação ISDL.

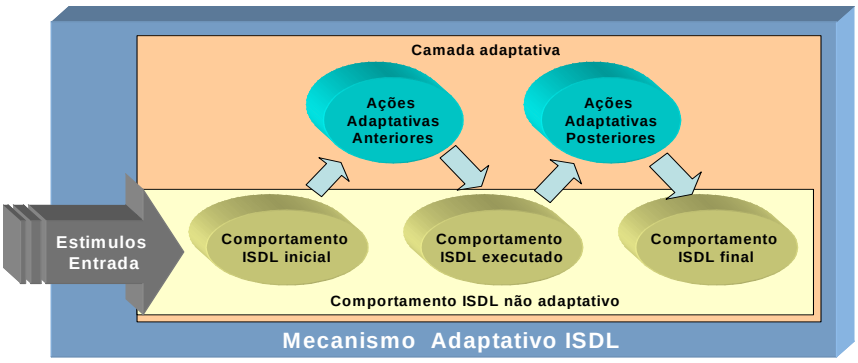


Figura 4 - Mecanismo Adaptativo ISDL.

Tomando-se como base o mecanismo adaptativo geral apresentado em [12], pode-se definir o modelo ISDL-Adp iniciando sua operação na configuração inicial c_0 na forma ISDL-Adp$_0$= (C_0, AR_0, Σ, c_0, A, NA, BA, AA). No passo k≥0, um estímulo de entrada movimenta ISDL-Adp para uma próxima configuração e então prossegue sua operação no passo k+1 se e somente se uma ação adaptativa não vazia for executada. Assim, estando ISDL-Adp em seu passo k, na forma ISDL-Adp$_k$ = (C_k, ARC_k, Σ, c_k, A, NA, BA, AA), a execução de uma ação adaptativa não-vazia o leva à forma ISDL-Adp$_{k+1}$ = (C_{k+1}, AR_{k+1}, Σ, c_{k+1}, A, NA, BA, AA). Nesta formulação:

- ISDL-Adp = (ISDL$_0$, AM) é um mecanismo adaptativo formado por um mecanismo inicial subjacente ISDL $_0$ e um mecanismo adaptativo AM;
- ISDL é o modelo subjacente não-adapatativo, cujo funcionamento foi descrito na seção 2, no passo k. ISDL $_0$ é o mecanismo subjacente, definido no conjunto inicial RC_0 (conjunto de ações ou relações de causalidades representando um comportamento não-adaptativo). Por definição, qualquer ação ou relação de causalidade não adaptativa em RC_k tem sua regra correspondente adaptativa em ARC_k;
- C_k é o conjunto de todos os possíveis comportamentos de ISDL no passo k, e $c_k \in C_k$ é o seu comportamento inicial no passo k. Para k=0, tem-se, respectivamente, C_0, o conjunto inicial de ações e relações de causalidade válidas e $c_0 \in C_0$, a configuração inicial de ISDL $_0$ e de ISDL.
- ε ("cadeia vazia") denota a ausência de qualquer outro elemento válido do conjunto correspondente;
- Σ é o conjunto (finito, fixo) de todos os possíveis eventos (inclusive o evento vazio ε) de que se compõe a cadeia de entrada AD;
- A ⊆ C é o subconjunto de ações e relações de causalidade de aceitação do comportamento ISDL;
- F = C - A é o conjunto das configurações de rejeição, ou seja, é o conjunto resultante da subtração das configurações de aceitação do conjunto de todas as possíveis configurações de comportamento;
- BA e AA são conjuntos de ações adaptativas. Ambas incluem a ação vazia (ε ∈ BA∩AA)
- w = w_1 w_2 ... w_n, é a cadeia de entrada, onde $w_k \in \Sigma - \{\varepsilon\}$, k =1, ...,n com n ≥ 0;
- NA, com ε∈NA, é um conjunto (finito, fixo) de todos os símbolos que podem ser gerados como saídas por ISDL-Adp, em resposta à aplicação de regras (ações e relações) adaptativas. Analogamente ao que ocorre com os mecanismos não adaptativos, a cadeia de saída assim obtida pode (se for conveniente) ser interpretada como uma sucessão de chamadas dos procedimentos correspondentes;
- ARC_k é o conjunto das regras adaptativas que definem o comportamento adaptativo de ISDL-Adp no passo K e é dado por uma relação $ARC_k \subseteq BA \times C \times \Sigma \times C \times ISDL \times AA$.

Em particular, ARC_0 define o comportamento inicial de ISDL-Adp. Ações adaptativas modificam o comportamento adaptativo corrente ARC_k de ISDL-Adp para um novo comportamento ARC_{k+1} adicionando e/ou eliminando inter(ações) e relacções de causalidade em ARC_k. Regras arc∈ARC_k são da forma:

<<ba>, ⟨⟨*a*, I, T, Λ, ς⟩, Γ, υ, ⟨*I-Refs*, *T-Refs*, *L-Refs*, *ITLRefs* ⟩, ⟨*I-Caus*, *T-Caus*, *L-Caus*, *ITL-Caus*⟩⟩, <aa>>

significando que, em resposta a algum símbolo da cadeia de entrada σ∈Σ, arc inicialmente executa a ação adaptativa ba∈BA. Se a execução de ba eliminar ar de ARC_k, a execução de arc é abortada; caso contrário, aplica-se a regra subjacente não-adaptativa:

- arc = ⟨⟨a, I, T, Λ, ς⟩, Γ, υ, ⟨I-Refs, T-Refs, L-Refs, ITLRefs ⟩, ⟨I-Caus, T-Caus, L-Caus, ITL-Caus⟩⟩ ∈ RC_k, conforme descrito anteriormente; e finalmente, executa-se a ação adaptativa aa∈AA.

Define-se ARC como o conjunto de todas as possíveis regras adaptativas para ISDL-Adp;
Define-se RC como o conjunto de todas as possíveis ações e relações de causalidade não-adaptativas para ISDL-Adp;

- AM ⊆ BA×RC×AA, definido para um particular mecanismo adaptativo ISDL-Adp, é um mecanismo adaptativo aplicado a todas as regras no passo k em $RC_k \subseteq RC$. AM deve ser interpretado da mesma forma como se fosse aplicada a qualquer sub domínio $RC_k \subseteq RC$. Isso determinará um único par de ações adaptativas associadas a cada regra não adaptativa.

O conjunto $ARC_k \subseteq ARC$ pode ser obtido colecionando-se todas as regras adaptativas construídas pela associação

de cada par de ações adaptativas às correspondentes regras não-adaptativas em RC_k.
O comportamento não-adaptativo $ISDL_i$, em cada passo de execução do mecanismo adaptativo, constitui o modelo ISDL-Adp por meio de suas ações, suas condições de causalidade básica, das relações entre ações. Uma ação ISDL-Adp pode ser representada por:

⟨⟨a, I, T, Λ, ς⟩ : 𝒜 , → ⟨⟨a`, I`,T`, Λ`, ς`⟩, Γ`, υ`, ⟨I-Refs`, T-Refs`, L-Refs`, ITLRefs` ⟩, ⟨I-Caus`, T-Caus`, L-Caus`, ITL-Caus`⟩⟩ : ℬ

Onde: ⟨⟨ a`, I`, T`, Λ`, ς`⟩, Γ`, υ`, ⟨I-Refs`, T-Refs`, L-Refs`, ITLRefs` ⟩, ⟨I-Caus`, T-Caus`, L-Caus`, ITL-Caus`⟩⟩ representa a situação do comportamento C_i antes da execução de uma ação adaptativa e é determinado de acordo com a definição anterior para comportamentos ISDL.
A Figura 5 ilustra a representação de uma ação ISDL-Adp genérica e suas funções adaptativas anteriores (𝒜) e posteriores (ℬ). Graficamente estas funções são representadas em caixas textos acopladas a parte superior e inferior das ações ISDL tradicionais.

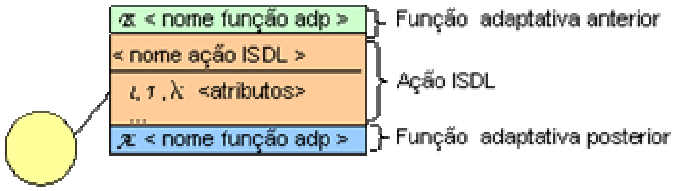


Figura 5 - Representação de ações adaptativas.

Visando diminuir a complexidade e facilitar a leitura das especificações ISDL-Adp geradas, as funções adaptativas são especificadas separadamente das ações ISDL-Adp. Textualmente, foram adicionadas as especificações ISDL tradicionais duas seções: *Before* e *After* que servem, respectivamente, para modelar as funções adaptativas anteriores e posteriores. No interior de cada seção são declaradas as funções adaptativas associadas a cada inter(ação). A Figura 6 ilustra a estrutura básica definida para as seções adaptativas anteriores e posteriores e suas respectivas funções e ações adaptativas.

```
.... outras seções ISDL-Adp
Before
        [<nome da função adaptativa anterior>] :
                (ação adaptativa elementar)
                ....
      ... (declaração de outras funções adaptativas)
After
        [<nome da função adaptativa posterior>] :
                (ação adaptativa elementar)
                ....
      ... (declaração de outras funções adaptativas
..... demais seções ISDL-Adp
```

Figura 6 – Representação de funções e ações elementares adaptativas ISDL.-Adp.

As ações adaptativas elementares ISDL-Adp são definidas de forma semelhante as ações adaptativas propostas para Autômatos Adaptativos [8]. As ações adaptativas ISDL-Adp elementares assumem o seguinte formato: *prefixo* [*padrão da produção*], onde *prefixo* representa um dos três tipos de ações adaptativas elementares a serem executadas: "**?**" (ação de inspeção), "**-**" (ação de eliminação) e "**+**" (ação de inserção), enquanto que o *padrão da produção* corresponde um nome da função que representa a ação ISDL tradicional. Desta forma temos uma ação adaptativa elementar representada da seguinte forma:

[prefixo] ⟨⟨a, I,T, Λ, ς⟩, Γ, υ, ⟨I-Refs`, T-Refs`, L-Refs`, ITLRefs` ⟩, ⟨I-Caus`, T-Caus`, L-Caus`, ITL-Caus`⟩⟩

onde: ⟨⟨a, I,T, Λ, ς⟩, Γ, υ, ⟨I-Refs`, T-Refs`, L-Refs`, ITLRefs` ⟩, ⟨I-Caus`, T-Caus`, L-Caus`, ITL-Caus`⟩⟩ faz parte do comportamento ISDL definido na seção 2 e o prefixo da ação adaptativa é definido conforme o tipo de uma ação adaptativa (prefixo), esta desempenha uma função que pode ter as seguintes características:

- ação adaptativa de inspeção: tem função de inspecionar/verificar se determinadas estruturas ISDL fazem parte da especificação do comportamento modelado.
- ação adaptativa de eliminação: tal estrutura é utilizada para eliminar estruturas ISDL necessárias as modificações que devem ocorrer no sistema modelado. Ao executar uma ação adaptativa de eliminação estruturas são eliminadas do comportamento ISDL
- ação adaptativa de inserção: serve para adição de novas estruturas ao comportamento ISDL. É por meio da execução de ações adaptativas de adição que novas estruturas são adicionadas ao comportamento ISDL

A operação do modelo ISDL-Adp ocorre por meio da modificação das ações pertencentes a um comportamento pela

evolução gradual do seu conjunto de regras, com a adição ou remoção de ações efetuadas por meio da execução das ações adaptativas.
Estando o ISDL-Adp em sua situação inicial, recebe uma seqüência de eventos externos. As transições são executadas conforme essa seqüência, consumindo os eventos externos, repetindo-se o processo até o final da seqüência. Para cada evento recebido verifica-se qual ou quais são as produções a serem executadas. No que se refere à execução do ISDL-Adp, se a ação adaptativa $\mathcal{A}$ estiver presente, será executada em primeiro lugar, enquanto que se $\mathcal{B}$ estiver declarada, será executada por último.

4. Estudo de Caso

Nesta seção será apresentado um exemplo que ilustrará a utilização do modelo e linguagem ISDL-Adp na modelagem de aplicações complexas. O exemplo escolhido para ser modelado é comum entre os usuários de computadores pessoais e de tecnologia Microsoft Windows [18] e tem por objetivo ilustrar o funcionamento das opções "Copiar", "Colar" e "Colar Especial" oferecidas na maioria dos produtos disponíveis para esta tecnologia. Na Figura 7 é ilustrado parte da interface da ferramenta Microsoft Office Word 2003 [19] onde aparecem desabilitadas as opções "Copiar", "Colar" e "Colar Especial" devido o conteúdo da área de transferência estar vazio e nenhum conteúdo ter sido selecionado.

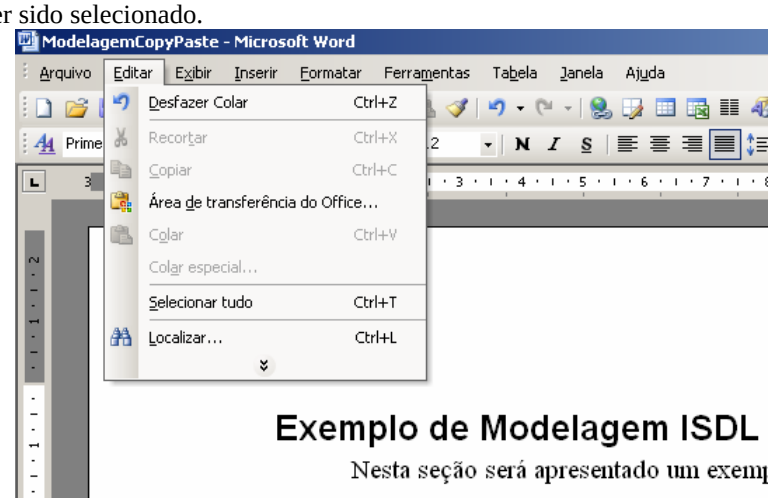


Figura 7- Situação inicial MS-Word Office XP 2003.

Para a opção "Colar" ser ativada faz-se necessário que o usuário selecione um texto, imagem ou algum outro objeto que torne a opção ativa conforme ilustrado na Figura 2.

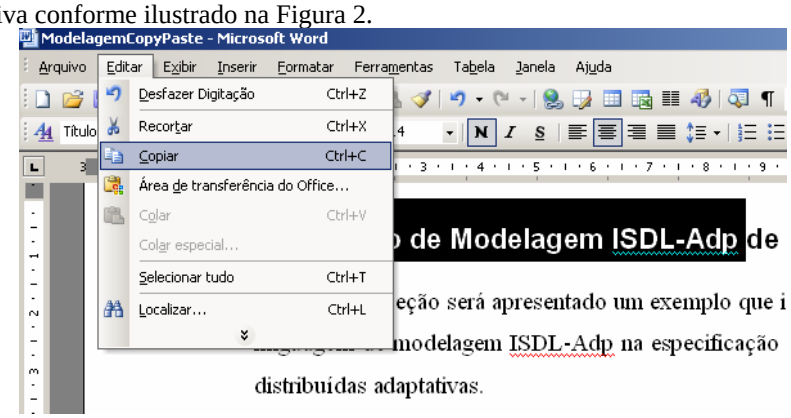


Figura 8 - Habilitação da opção "Copiar".

Ao ser selecionada a opção "Copiar" ou ativada a mesma por teclas de atalho é copiado o conteúdo selecionado para a Área de Transferência e habilitada a ocorrência das opções "Colar" e "Colar Especial" do menu de opções. Pode-se observar que ao ser selecionado um texto e copiado o seu conteúdo para a área de transferência são atribuídas a operação de colar "Texto Formatado" para a opção "Colar" e as operações de colar "Texto unicode sem fortação", "Formato Html", "Texto Formatado", "Texto sem Formatação", "Texto Formatado RTF", etc. para a opção "Colar Especial" Tais opções são habilitadas conforme o conjunto de programas instalados anteriormente que permitiram que determinadas operações fossem disponibilizadas. A Figura 3 ilustra a interface que é apresentada após a escolha da opção "Colar Especial". A interface habilitada apresenta ao usuário uma lista de possíveis operações para serem realizadas conforme sua necessidade e escolha.

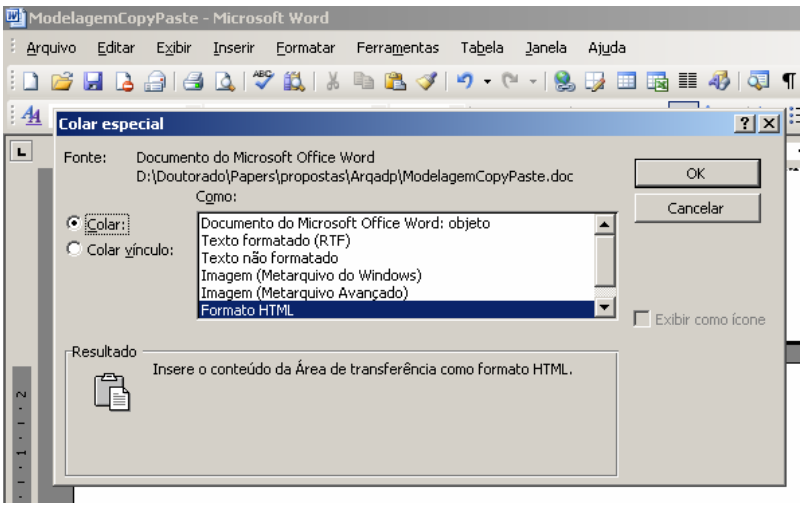


Figura 9 - Operações disponíveis ao escolher a opção "Colar Especial" para objetos do tipo Texto.

Em relação a cópia do conteúdo de uma imagem para a área de transferência ocorre situação semelhante a que ocorre ao ser copiado um texto. Ou seja, ao ser selecionada uma imagem é executada a ação de cópia de conteúdo para a área de transferência disponível na opção "Copiar" e atribuídas as operações "Colar Bitmap" para a opção "Colar" e as operações: "Bitmap", "Imagem (Meta Arquivo Avançado)", "Imagem GIF", etc. para a opção "Colar Especial". Da mesma forma que para a opção de cópia de um texto as operações são disponibilizadas conforme o conjunto de programas instalados anteriormente no equipamento, o tipo de imagem selecionado e a origem do conteúdo a ser copiado para a área de transferência (programa em que foi gerada a cópia). Desta forma pode-se observar que o conjunto de operações disponibilizadas pelo comando "Colar Especial" pode ser alterado em tempo de execução e disponibilizar diferentes operações conforme características da origem e do conteúdo copiado. Tal situação ilustra a aplicação de tecnologia adaptativa e demonstra o seu uso na prática. Vale ressaltar que este trabalho não tem foco principal na modelagem de características das opções "Copiar", "Colar" e "Colar Especial" e sim na aplicação de tecnologia adaptativa na modelagem de aplicações complexas. Outras opções de cópias de conteúdos como imagem tipo vetorial, áudio, vídeo, etc.. são sugeridas para efeitos de estudos futuros.

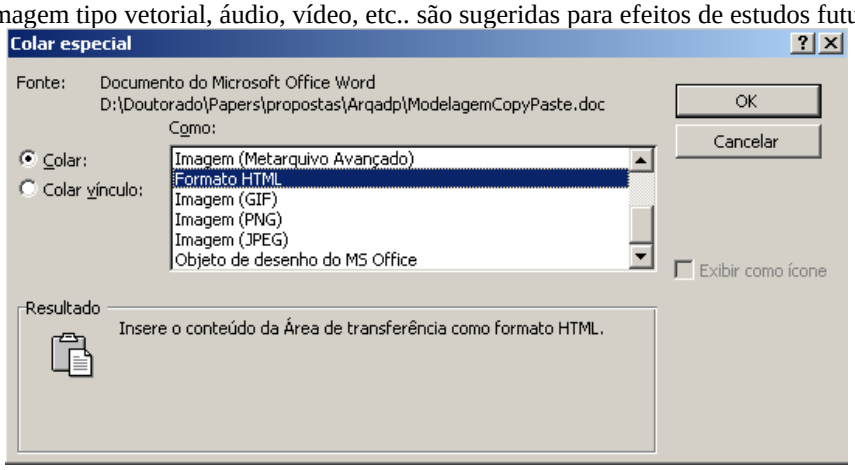


Figura 10 - Operações disponíveis ao escolher a opção "Colar Especial" para objetos tipo Imagem.

4.1 Modelagem ISDL-Adp de Copiar/Colar conteúdo Área de Transferência.

Fundamentado no problema descrito anteriormente é apresentada a modelagem ISDL-Adp de Copia/Colar conteúdo Área Transferência. A Figura 11 representa a modelagem de entidades do *Copiar/Colar conteúdo Área Transferência* que é representada pela na visão externa por um *Usuário* que interage com um *Sistema de Computação*. Na visão interna do sistema de computação foram abstraídos alguns detalhes visando simplificar a sua apresentação. Desta forma temos o *Sistema de Computação* composto por três subentidades: *Interface da Aplicação*, *Sistema Operacional* e *Área de Transferência*. A *Interface da Aplicação* tem por finalidade receber interações vindas do usuário e enviar ao sistema operacional objetivando a sua execução. Esta entidade também recebe mensagens vindas do *Sistema Operacional* e da *Área de Transferência* e as apresenta aos usuários. A subentidade *Sistema Operacional* tem por objetivo realizar o gerenciamento do *Sistema de Computação* e a subentidade *Área de Transferência* é um espaço de memória que permite ao usuário realizar cópias de dados ao utilizar suas aplicações ou intercambiar dados entre as aplicações utilizadas.

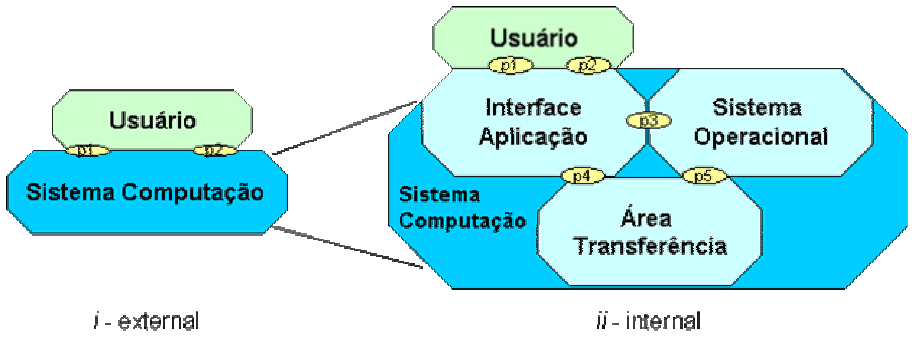


Figura 11 - Modelo Entidades Exemplo Copiar-Colar Área Transferência.

A subentidade *Interface da Aplicação* possui diversas funcionalidades conforme a aplicação que é executada. Na Figura 12 é ilustrada parte do comportamento *C_Copiar_Colar* . Tal comportamento representa as funcionalidades de Copiar e Colar conteúdo para Área de Transferência e é comum para a maioria das aplicações existentes. O comportamento ilustrado representa o configuração inicial da aplicação e constitui-se da interação *i_Cop* que tem por objetivo modelar o recebimento das interações vindas do usuários referentes a requisição de cópia de um determinado conteúdo selecionado para a área de transferência. Ao ser ativada a interação *i_Cop* ocorre a habilitação das ações *aCopTxt* e *aCopImg* que possuem a função adaptativa anterior *FReset(str)* responsável por restaurar o comportamento em execução para a situação inicial, caso este tenha sido adaptado anteriormente, e a função adaptativa posterior *FCop(str,str)* responsável por realizar mudanças no comportamento atual conforme o conteúdo selecionado pelo usuário.

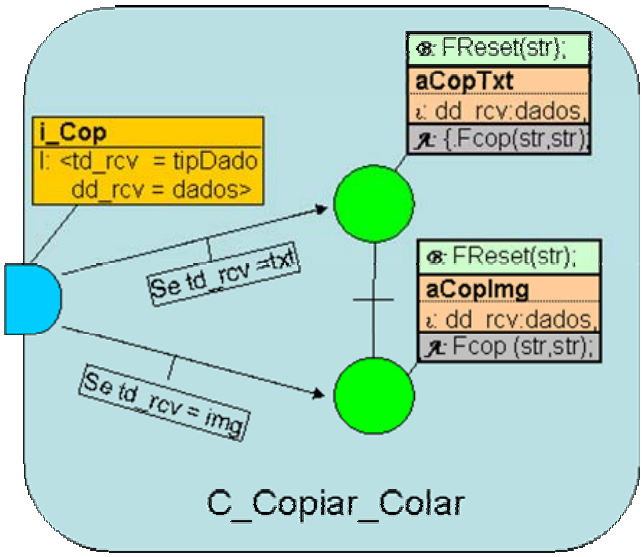


Figura 12 - Comportamento inicial funcionalidade Copiar/Colar conteúdo para Área de Transferência.

Na Figura 13 são apresentados os subcomportamentos *C_Cop_Txt* e *C_Cop_Img*. O subcomportamento *C_Cop_Txt* é responsável por adicionar ou remover do comportamento inicial funcionalidades relacionadas a cópia do conteúdo do tipo texto. Este subcomportamento ao receber uma mensagem oriunda da interação *icl* no ponto de entrada *e1*, referente a colar o conteúdo da área de transferência, habilita a ocorrência da ação *aCl_TxtFmt* responsável por colar o conteúdo da área de transferência como sendo um texto formatado. Ao receber um mensagem no ponto de entrada *e2*, oriunda da interação *icle* (colar especial), ocorre a habilitação da escolha de ocorrência de uma das ações *aCl_TxtFmt*, *aCl_TxtSemFmt* e *aCl_TxtHtml* responsáveis respectivamente por colar textos formatados, sem formatação ou no formato HTML. Após execução de uma das possíveis ações é enviada mensagem por meio do ponto de saída *s1* contendo informações para serem apresentadas ao usuário por meio da interação *i_apu.*
O subcomportamento *C_Cop_Img* é constituído de forma semelhante ao comportamento *C_Cop_Txt* diferenciado-se no conjunto de ações a serem desempenhadas que estão relacionadas a operação de colar imagens. A ação *ACl_ImgBmp* é responsável por colar imagens do tipo *Bitmap* e é padrão para a interação *icl*. As ações *aClImgBmp*, *aCl_ImgMtArAvc* e *aCl_ImgGif* são responsáveis respectivamente por colar imagem do tipo *Bitmap*, *Meta Arquivo Avançado* e *Gif* estão relacionadas a interação *icle* e ocorrem em escolha.

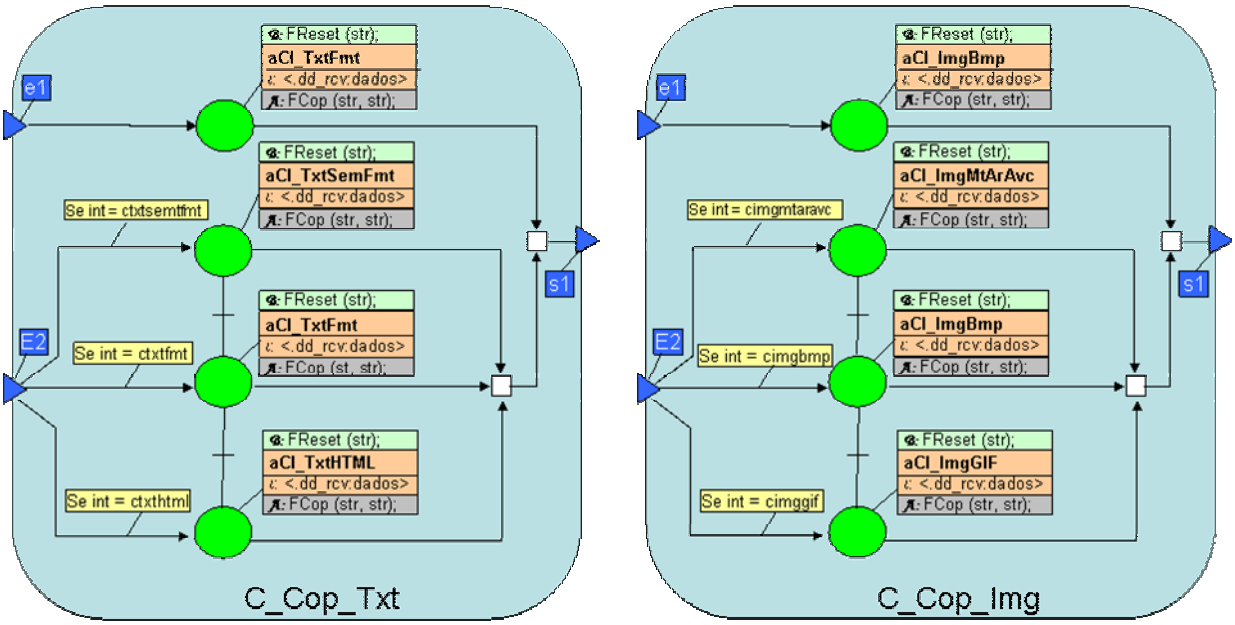


Figura 13 – Subcomportamentos *C_Cop_Img* e *C_Cop_Txt* .

Na Figura 14 são apresentadas as funções adaptativa *FCop(str, str)* e *FReset(str)*. Tais funções são responsáveis por realizar mudanças no comportamento corrente em um determinado instante de execução. A função *Fcop(str,str)* recebe dois parâmetros referente ao nome do comportamento inicial e comportamento que será inserido (adaptado). A ser chamada esta função ocorre a execução de um conjunto de ações adaptativas elementares que tem por objetivo inserir um novo subcomportamento que deverá desempenhar a função de colar conteúdo da área de transferência. Após a inserção do novo subcomportamento ocorre a ligação de seus pontos de entrada e de saída com as respectivas interações do comportamento inicial de forma que a aplicação possa disponibilizar as novas funcionalidades inseridas em seu conjunto de regras. A função *FReset (str)* executa inicialmente uma ação elementar de consulta e verifica no conjunto de regras a existência de uma regra que atenda as condições exigidas. Na existência da regra, a variável *adaptador* recebe o nome do respectivo subcomportamento que atende a referida consulta. Após ser definido qual subcomportamento pertence a configuração atual este e suas ligações de entrada e de saída devem ser removidas de forma a aplicação retornar a sua situação de configuração inicial.

FCop (*inicial*: char, *adaptatdor*: char); **{** + *adaptator*; + icl.*inicial* ← e1.*adaptador*; + icle.*inicial* ← e2.*adaptador*; + s1.*inicial* ← i_apu.*adaptador*; + aCopTxt.*inicial* ∨ aCopImg.*inicial* ← icl.*inicial*; + aCopTxt.*inicial* ∨ aCopImg.*inicial* ← icle.*inicial*; **}**	***FReset*** **(***inicial*: char**)** ***var*** *adaptador*:char; **{** ? icl.*inicial* ← e1.*adaptador*; - icl.*inicial* ← e1.*adaptador*; - icle.*inicial* ← e2.*adaptador*; - s1.*adaptador* ← i_apu.*adaptador*; - aCopTxt.*inicial* ∨ aCopImg.*inicial* ← icl.*inicial*; - aCopTxt.*inicial* ∨ aCopImg.*inicial* ← icle.*inicial*; - *adaptador*; **}**

Figura 14 - Funções Adaptativas *FCop(...)* e *FReset(...)*.

Na Figura 15 é apresentado o comportamento *C_Copiar_Colar* que foi modificado ao ser habilitada a execução da ação *aCopImg* e suas respectivas ações adaptativas anterior e posterior. Anteriormente a execução de *aCopImg* foi executada a função adaptativa *FReset(str)* que restaurou o comportamento para a configuração incial, na seqüência ocorreu a execução da ação *aCopImg* que copiou o conteúdo selecionado para a área de transferência e após a execução de *aCopImg* foi executada a ação adapatativa *FCop(str, str)* que realizou mudanças no comportamento incial adicionando novas funcionalidades e permitindo a realização de colar conteúdo da área de transferência do tipo imagem.

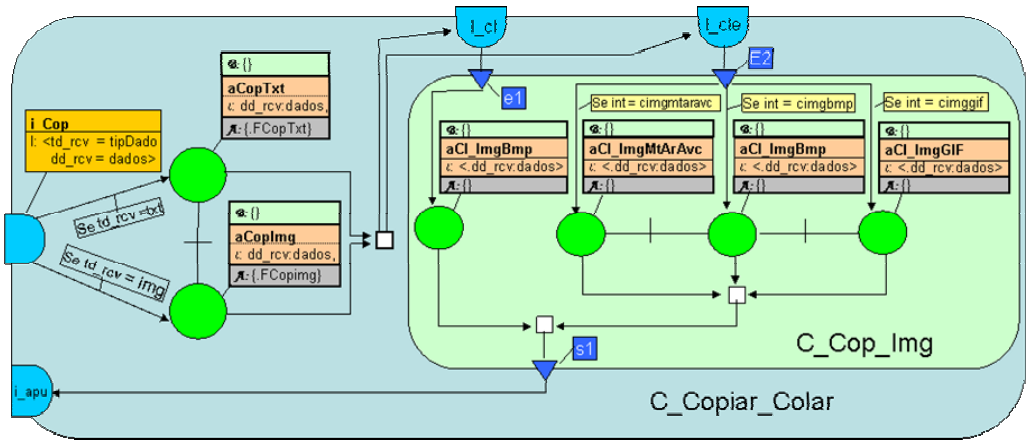


Figura 15 - Comportamento Comp_Copiar_Colar após execução ação Adaptativa.

5. Conclusão

Neste trabalho foi apresentada a estrutura básica do Modelo ISDL-Adp que foi projetado para a especificação de sistemas com características distribuídas e reconfiguráveis. Tal modelo é fruto da união dos conceitos do modelo não-adaptativo ISDL aos conceitos de mecanismos adaptativos.

No modelo ISDL-Adp, de forma análoga ao que ocorre com o modelo ISDL, uma entidade modela um objeto e é caracterizada por seu comportamento (métodos) e estado (atributos). Isso permite, por exemplo, que se estruture um sistema como uma hierarquia de abstrações (entidades) e que se encapsule o comportamento de entidades, pois estes somente são acessíveis por meio dos pontos de interação definidos em sua interface.

Por sua vez, a definição de ações adaptativas na estruturação de comportamentos permite o uso da adição e da otimização de ações e relações de causalidade, de forma dinâmica, o que torna o modelo ISDL mais flexível e com maior poder de expressão para problemas que apresentam características reconfiguráveis. Utilizando-se do modelo ISDL-Adp o projetista pode modelar comportamentos que tenham sua estrutura modificada durante a execução do sistema. Os recursos adaptativos permitem que sejam inseridas ou eliminadas ações e relações de causalidade a cada execução do comportamento, o que permite que aplicações com grande complexidade de representação sejam expressas com o auxílio deste modelo.

O exemplo ilustrativo fundamentado na modelagem de funcionalidade de copia/colar conteúdo para a área de transferência serviu para ilustrar a utilização do modelo no projeto de tais aplicações. Tal exemplo permitiu verificar que o modelo permite representar aplicações que necessitam realizar mudanças em seu comportamento durante o tempo de execução. A modelagem também permitiu que o projeto fosse realizado de forma mais organizada e que somente os recursos necessários sejam utilizados a cada execução. Dessa forma ocorreu uma otimização dos recursos e um melhor desempenho e utilização dos dispositivos móveis.

Como continuação deste trabalho deverá ser investigadas a possibilidade de utilização de conceitos de tecnologia adaptativa na modelagem de interações entre comportamento e na estrutura do modelo de entidades. Também faz-se necessário a utilização de uma metodologia para dar suporte no projeto de aplicações que possuem características adaptativas. Tal metodologia deverá ser fundamentada no modelo ISDL-Adp e ferramentas deverão ser construídas para dar suporte a esta metodologia. Também serão realizados estudos referentes a utilização do modelo no projeto de outros tipos de aplicação.

Referências Bibliográficas

[1] OMG. UML Profile for Enterprise Distributed Object Computing Specification, ptc/02-02-05, (2002).

[2] Harel, D., Pnueli, A., Schmidit, J.P. and Sherman, R.,. On the formal semantics of statecharts, In: Symposyum on logic in Computer Science, 2°, Ithaca, *Proceedings, IEEE Press, New York*, p.p.54-64, (1984).

[3] ISO IS 8807, Information Processing Systems - Open Systems Interconnection - LOTOS - A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour, *ISO*, 1989.

[4] ISO/IEC 9074 (1989) Information Processing Systems - Open System Interconnection - Estelle - A Formal Description Technique Based on an Extended State Transition Model, *ISO*, (1989).

[5] ITU-T "Recommendation Z.100", White Book and Annexes, 1994

[6] Pires L.F. Archictetural Notes: A Framework for Distributed Systems Development, *Ph.D Thesis Twente University, Netherlands*, (1994).

[7] Quartel, D. Actions Relations – Basic design concepts for behaviour modelling and refinement. *Ph.D Thesis Twente University, Netherlands*, (1997).

[8] Neto, J.J. Contribuição à metodologia de construção de compiladores. *Tese de Livre Docência, Escola Politécnica, Universidade de São Paulo*, São Paulo, (1993).

[9] Rady, J. STAD-Uma ferramenta para representação e simulação de sistemas através de statecharts adaptativos, *Tese de Doutorado, Escola Politécnica, Universidade de São Paulo*, São Paulo, (1995).

[10] Allen, R.J., Douence, R. and Garlan, D. Specifying and Analyzing Dynamic Software Architectures. *Proceedings of the 1998 Conference on Fundamental Approaches to Software Engineering* Lisboa, Portugal, (1998).

[11] Revault N. and Yoder, J. Adaptive Object-Models and Metamodeling Techniques *ECOOP '2001 Workshop Reader; Lecture Notes in Computer Science; Springer Verlag*, (June 2001).

[12] Neto, J.J. Adaptive rule-driven devices - general formulation and case study, *CIAA 2001 - Sixth International Conference on Implementation and Application of Automata, Pretoria University,* Pretoria-África do Sul, (2001).

[13] Sinderen M.van, Pires, L.F., Vissers, C.A.and Katoen, C.A A design model for open distributed processing systems. *Computer Networks and ISDN Systems*, (1995), pp. 1263-1285.

[14] Testbed Studio Tool, http://www.bizzdesign.com. (2004).

[15] Friends Project.http://www.telin.nl/Middleware/FRIENDS/ENindex.htm. (Maio 2004)

[16] Camolesi, A.R. and Neto, J.J. An adaptive model for modelling of distributed system, *Conference Argentina in la Ciência da Computacion (CACIC)*, La Plata, Argentina, (Outubro 2003).

[17] Camolesi, A.R. Uma metodologia para o Design de Serviços de TV-Interativa, *Dissertação de Mestrado, PPG-CC, UFSCar*, (Fevereiro 2000).

[18] Microsoft Windows XP disponível em: http://www.microsoft.com/brasil/windows/default.asp

[19] Microsoft Office Word 2003 disponível em http://www.microsoft.com/brasil/office/editions/standard.asp

HERRAMIENTA SOFTWARE CON INTERFAZ WEB PARA LA INTERPRETACIÓN SIMBÓLICA DE MODELOS NEURONALES

Denis Rincón
Escuela de Computación. Facultad de Ciencias.
Universidad Central de Venezuela. Caracas, Venezuela

Ely Rozo
Escuela de Computación. Facultad de Ciencias.
Universidad Central de Venezuela. Caracas, Venezuela

Haydemar Núñez
Laboratorio de Inteligencia Artificial
Escuela de Computación. Facultad de Ciencias.
Universidad Central de Venezuela. Caracas, Venezuela
hnunez@strix.ciens.ucv.ve

Resumen.
En este trabajo se presenta el Sistema Interpretador de Modelos Neuronales WebTREPAN. Esta herramienta Web permite extraer, de manera automática, una representación simbólica aproximada en forma de un árbol de decisión, del conocimiento que ha capturado una red neuronal durante su aprendizaje. La herramienta está basada en el algoritmo de extracción de reglas TREPAN y mediante una interfaz guiada tipo "Wizard", el usuario introduce los datos relacionados con la red neuronal que desea interpretar (tales como la topología de la red neuronal). De esta forma, WebTREPAN puede ser utilizado para facilitar la comprensión y el análisis del proceso que está bajo investigación, servir de apoyo para la toma de decisiones por parte de expertos y complementar otros sistemas utilizados en la minería de datos.
Palabras claves: Inteligencia Artificial, Redes Neuronales, Extracción de Reglas, Minería de Datos

Abstract
In this work WebTREPAN a Neural Networks Interpreter Systems is presented. This Web tool allows extract a approximate symbolic representation (in form of a decision tree) from trained neural networks automatically. WebTREPAN is based on TREPAN, a global rule extraction algorithm. It have a wizard interface, where the user can introduce the neural network data (as such the topology). In this way, it is possible to give an interpretation to the knowledge acquired by a neural networks during its learning. Therefore, WebTREPAN can be used for data exploration and for data mining applications.
Keywords: Artificial Intelligence, Neural Networks, Rule Extraction, Data Mining

1. INTRODUCCIÓN

Las redes neuronales se han utilizado ampliamente en la resolución de problemas complejos en diferentes dominios, donde han mostrado buenas capacidades de generalización [4],[10],[13]. Sin embargo, una posible limitación de estas máquinas de aprendizaje es que generan modelos del tipo *caja negra*; es decir, la solución suministrada por ellas es difícil de interpretar desde el punto de vista del usuario [24]. Convertir en interpretable estos modelos, ha sido el objetivo de diferentes métodos desarrollados en los últimos años que permiten extraer reglas a partir de redes neuronales entrenadas [1],[8],[21],[22],[23].

Además de dotar a las redes neuronales de la *capacidad de explicación*, estos *métodos de extracción de reglas* posibilitan que los modelos obtenidos sean utilizados para la adquisición automática de conocimiento en sistemas expertos [1],[17]. También, la extracción de reglas mejora la adecuación de las redes neuronales para resolver problemas de minería de datos ("data mining") [2],[8],[16], cuando el objetivo principal es descubrir relaciones desconocidas e implícitas en grandes bases de datos que, en muchos casos, es necesario expresarlas en un formato comprensible.

Uno de los métodos de extracción de reglas actuales es el algoritmo TREPAN desarrollado por Craven [7], [9]. Este algoritmo realiza la extracción desde redes neuronales entrenadas y obtiene representaciones simbólicas en forma de árboles de decisión. Una característica resaltante de TREPAN es que al ser un método de extracción global, puede aplicarse a diferentes tipos de redes neuronales y en general, a cualquier paradigma de aprendizaje supervisado en entornos de clasificación.

En este trabajo se presenta el Sistema Interpretador de Modelos Neuronales WebTREPAN, una herramienta automatizada basada en Web que incorpora como núcleo el método de extracción de reglas TREPAN. Este sistema puede ser utilizado para trasladar de manera automática un alto porcentaje del conocimiento sintetizado en una red neuronal entrenada del tal forma que ese conocimiento, expresado en la nueva representación en forma de un árbol de decisión, pueda ser utilizado para facilitar la comprensión y el análisis del problema que está bajo investigación.

Este artículo está organizado de la siguiente forma: En la sección 2 se presentan brevemente los fundamentos de los métodos de extracción de reglas a partir de redes neuronales entrenadas. En la Sección 3 se describe el método TREPAN para la extracción de reglas. A continuación, en la Sección 4, se presenta el sistema WebTREPAN, donde se describen los módulos de interfaz y de interpretación. En la Sección 5 se muestran las pruebas realizadas para verificar la funcionalidad de WebTREPAN. Por último, se presentan las conclusiones y se comentan los trabajos futuros.

2. REDES NEURONALES Y LA EXTRACCIÓN DE REGLAS

Las redes neuronales artificiales [4],[10],[13] representan, más que una sola técnica de aprendizaje, una familia de modelos que han sido utilizados con éxito en una amplia variedad de problemas de clasificación y de regresión, donde han exhibido las siguientes prestaciones: un alto rendimiento de generalización ante ejemplos no vistos durante el aprendizaje, robustez de la solución ante la presencia de imprecisión, ruido o incertidumbre en los datos de entrada, y habilidad para aprender relaciones no lineales, la cual es una característica importante para manipular bases de datos reales. Sin embargo, en los modelos neuronales el conocimiento derivado del proceso de aprendizaje se encuentra sintetizado en una forma sub-simbólica (matrices de pesos) que hace difícil su interpretación y por lo tanto, son mecanismos de resolución de problemas tratados como una *caja negra*.

Para superar la limitación de las redes neuronales relacionada con su *falta de transparencia*, la hipótesis generada por estos modelos podría trasladarse a una forma interpretable; los métodos que realizan esta transformación se conocen como *algoritmos de extracción de reglas* [1],[8],[21],[22],[23]. Y, aunque el requerimiento de una solución comprensible dependerá de las necesidades de la aplicación y del usuario final del sistema, las ventajas que se obtienen con esta transformación son:

- *Provisión de la capacidad de explicación*, lo cual redundará en un mejor entendimiento del problema bajo estudio y de su solución. Esto es especialmente importante cuando la solución final será utilizada para procesos de toma de decisiones.
- *Exploración de datos*, al *revelar* las relaciones entre las variables del sistema que hayan sido descubiertas durante el aprendizaje y que se encuentran codificadas en un formato incomprensible [3].

La extracción de reglas también posibilita que las redes neuronales puedan utilizarse como una herramienta para la adquisición automática de conocimiento en sistemas basados en reglas [1],[17]. El mayor inconveniente a la hora de diseñar un sistema experto está relacionado con la construcción y depuración de su base de conocimiento. Las redes neuronales, junto con la extracción de reglas, podrían ayudar en este sentido al realizar un aprendizaje a partir de ejemplos representativos del problema y luego, las reglas extraídas pasarían a formar parte de la base de conocimiento.

También estos métodos de extracción de reglas mejoran la adecuación de las redes neuronales para aplicaciones de minería de datos ("data mining") [1],[8],[16] cuando en muchos casos se requiere que las relaciones que hayan sido descubiertas en grandes bases de datos estén expresadas en un lenguaje comprensible.

2.1 Características de los Métodos de Extracción de Reglas

El conocimiento capturado por una red neuronal durante su aprendizaje está representado en términos de la topología de la red, las funciones de activación utilizadas por las neuronas, y los parámetros asociados con las conexiones (pesos) y las unidades (umbrales) [1],[8]. La técnica de extracción debe entonces interpretar estos tres elementos y, a partir de ellos, generar una descripción más comprensible que sea funcionalmente equivalente a la red neuronal (es decir, que suministre las mismas respuestas de predicción).

Los diversos métodos que han sido desarrollados para extraer y representar el conocimiento integrado en una red neuronal, pueden distinguirse de acuerdo a las siguientes características:

- *Lenguaje de representación* utilizado por el método para describir el modelo aprendido por la red neuronal. Los lenguajes que han sido utilizados son: reglas del tipo *si-entonces*, reglas *m-de-n*, árboles de decisión, autómatas de estado finito, reglas oblicuas (funciones lineales), reglas difusas ("fuzzy), entre otras.
- *La estrategia de extracción*, la cual define la manera en que el método utiliza a la red para realizar la extracción. En este sentido se distinguen métodos *locale*s, que analizan la estructura de la red a nivel de las unidades ocultas y de salida para extraer las reglas, y métodos *globales*, que extraen reglas sobre la base de la función entrada-salida implantada por la red, sin analizar su estructura interna. También se han propuesto técnicas *híbridas* que se encuentran entre estos dos extremos.
- *Requerimientos sobre la red*. Algunos algoritmos sólo son aplicables a ciertas arquitecturas de redes y/o pueden requerir de características específicas tales como ciertos tipos de neuronas o patrones de interconexión. Además, pueden necesitar de regímenes especiales de entrenamiento con el fin de facilitar el proceso de extracción. Otros métodos son independientes de la arquitectura o pueden ser aplicados a diferentes redes. De alguna manera, esta característica define la *portabilidad* del algoritmo de extracción.

Para finalizar, el rendimiento y calidad del conjunto de reglas extraído puede valorarse utilizando las siguientes medidas [1],[16]: la *consistencia* o *equivalencia*, que representa la capacidad de las reglas para imitar el comportamiento de la red a partir de la cual fueron generadas; la *exactitud*, la cual es la habilidad del conjunto de reglas para clasificar correctamente los datos; y la *complejidad*, medida en términos del número de reglas extraídas y del número de antecedentes por regla.

3. EL ALGORITMO DE EXTRACCIÓN DE REGLAS TREPAN

El algoritmo TREPAN es un método global de extracción de reglas desarrollado por Craven [7],[9], que permite obtener una representación simbólica aproximada en forma de un árbol de decisión del concepto representado por una red neuronal entrenada. TREPAN utiliza un *Oráculo*, un sistema a quien dirigir preguntas y que sea capaz de clasificarlas de acuerdo al conocimiento que posea (en este caso la red neuronal). De esta forma, TREPAN puede aplicarse a diferentes paradigmas de aprendizaje supervisados.

Existen diversos algoritmos de inducción de árboles de decisión a partir de un conjunto de ejemplos, como el ID3 o C4.5 [15]. Básicamente, TREPAN es similar a ID3 (ver Figura 1) con algunas diferencias que se especifican brevemente a continuación:

El Oráculo

El objetivo del algoritmo TREPAN es extraer y representar el concepto que aprendió la red neuronal, por esta razón utiliza a la red como un Oráculo para clasificar cada dato que se le presenta. Es así que lo primero que hace TREPAN es re-clasificar con el Oráculo a todos los datos que serán utilizados para derivar el árbol de decisión, buscando que la representación obtenida sea lo más fiel posible a ese concepto.

Generación de nuevas instancias

Otra característica de TREPAN es su capacidad para generar nuevas instancias, las cuales son clasificadas por el Oráculo. De esta forma, TREPAN asegura que se mantenga siempre una *cantidad mínima* de datos en el nodo antes de obtener la etiqueta de clase (si es una hoja) o seleccionar un test de partición (si es un nodo interno del árbol).

Para llevar a cabo este proceso, TREPAN construye un modelo de la distribución de los datos que llegan al nodo y luego usa este modelo para generar las nuevas instancias. Estos modelos toman en cuenta el conjunto de restricciones que causan que un dato siga un camino desde el nodo raíz hasta el nodo donde se desea generar las instancias. Estas restricciones pueden ser de dos tipos: no-disyuntivas y expresiones *m-de-n*.

Expansión del Árbol

Para realizar la expansión de un nodo, TREPAN hace uso del criterio *primero el mejor,* en donde primero se realiza la expansión del nodo con mayor potencial de incrementar la fidelidad del árbol que es extraído de la red. La función usada para evaluar un nodo viene dada por la siguiente expresión:

$$Evaluar(Nodo) = Alcanzar(Nodo)*(1 - Fidelidad(Nodo)) \quad (1)$$

```
ALGORITMO TREPAN (Entradas : datos de entrenamiento S, variables F, criterios de parada,
Parámetro mínimo_datos)

Cola := { vacía };      /* se mantiene una cola de nodos para posibles expansiones */

        PARA cada dato E del conjunto S   /* usar la red para clasificar los datos de entrenamiento.*/
1.-     Asignar una etiqueta de clase para el dato E := CONSULTAR_AL_ORÁCULO(E)
        FPARA

2.-     Inicializar la raíz del árbol T, como un nodo hoja
3.-     Construir el Modelo de Distribución M de los datos que alcanzan el nodo T
4.-     query_instances := TRAZAR_EJEMPLO ( { }, mínimo_datos - |S|, M )
5.-     Usar S y las query_instances para determinar la etiqueta de clase para el nodo T
6.-     Inicializar la Cola con la Tupla (T, S, query_instances_T , { })

        MIENTRAS (la Cola no este vacía) Y (el Criterio de Parada Global no este satisfecho)
            /* Se procede a Expandir el mejor nodo en base al criterio de evaluación*/
7.-         Sacar de la Cola el nodo N con (N, S_N, query_instances_N , restricciones_n)
8.-         Test:= CONSTRUIR_TEST(F, S_N U query_instances_N)   /* mejor partición binaria */
9.-         Hacer de N un nodo interno con el Test obtenido

10.-        PARA cada salida t del Test                  /* hacer los nodos hijos */

11.-            Hacer a C, un nuevo nodo hijo de N
12.-            RestriccionesC := restriccionesN U { Test = t }
13.-            Sc :=miembros de S_N con salida t en el Test
14.-            Construir el Modelo de Distribución M de los datos cubiertos por el nodo C
15.-            query_ instancias:= TRAZAR_EJEMPLO( restriccionesC, mínimo_datos-|Sc|, M )
16.-                                                          Usar Sc y las query_ instancesc para deter
inar la etiqueta de clase para el nodo C

17.-            SI (el Criterio de Parada Local no está satisfecho) ENTONCES
18.-                Poner {C, examplesC, constraintsC} dentro de la Cola, para futuras expansiones
                FSI

            FPARA
        FMIENTRAS
```

Figura 1. Algoritmo TREPAN para la extracción de reglas

donde, *Alcanzar(Nodo)* se refiere a la fracción estimada de datos que llegan al nodo, y *Fidelidad(Nodo)* se refiere a la fracción de instancias para las cuales el árbol y la red coinciden en sus predicciones.
TREPAN mantiene una cola de nodos para futuras expansiones y en base a esta función de evaluación, determina cuál es el mejor nodo que será sacado de la cola para su expansión. Es importante mencionar que por cada nodo en la cola, TREPAN mantiene los datos de entrenamiento que llegaron a él, las nuevas instancias que fueron creadas y las restricciones del mismo.

Selección de los Tests de Partición

TREPAN sólo utiliza tests binarios para realizar la partición del árbol. Estos tests pueden ser de dos tipos: no-disyuntivos y expresiones *m-de-n*. Para construir un test se consideran todos los posibles valores que tomen las variables tanto en los datos de entrenamiento que llegan al nodo como en las instancias creadas y clasificadas por el Oráculo. Por otra parte, TREPAN utiliza una medida de evaluación heurística llamada *criterio de ganancia de información* [15] para seleccionar el mejor test de partición (el que maximice este criterio de evaluación) sobre un conjunto de candidatos.

Criterios de Parada

Otro aspecto clave de los algoritmos de inducción de árboles de decisión, es determinar cuándo se debe detener el crecimiento del árbol. Existen varios criterios para decidir si se hace hoja a un nodo; TREPAN utiliza un criterio de parada local y otro global. El primero toma en cuenta la pureza de los datos que llegan al nodo, haciéndolo una hoja si todos los datos que llegan a él pertenecen a la misma clase. Por otro lado, el criterio global se refiere al tamaño máximo que debe tener el árbol que TREPAN va a retornar. Este parámetro puede ser proporcionado por el usuario y está especificado en términos de la cantidad de nodos internos, lo cual le da un control sobre la comprensibilidad que tendrán los árboles producidos por el algoritmo. Asimismo, TREPAN es capaz de usar el conjunto de datos de test (en caso de no especificarse un límite para el tamaño del árbol), para tomar la decisión sobre el tamaño el árbol que será devuelto. Este conjunto se utiliza para calcular la fidelidad de cada subárbol generado por el algoritmo a medida que el árbol va creciendo.

4. INTERPRETACIÓN AUTOMÁTICA DE REDES NEURONALES ENTRENADAS

El objetivo de este trabajo fue el desarrollo de un sistema automatizado para la interpretación de modelos neuronales basado en el algoritmo de extracción de reglas TREPAN. Esta herramienta, llamada WebTREPAN, permite extraer una representación explícita del conocimiento sintetizado en una red neuronal entrenada de forma automática. De esta forma, WEbTREPAN puede facilitar la labor de análisis del proceso o sistema que está bajo investigación, así como ayudar en la toma de decisiones por parte de una persona experta. También, esta herramienta podría complementar a los sistemas utilizados en la minería de datos. En la Figura 2 se muestra el modelo general del sistema WebTREPAN.

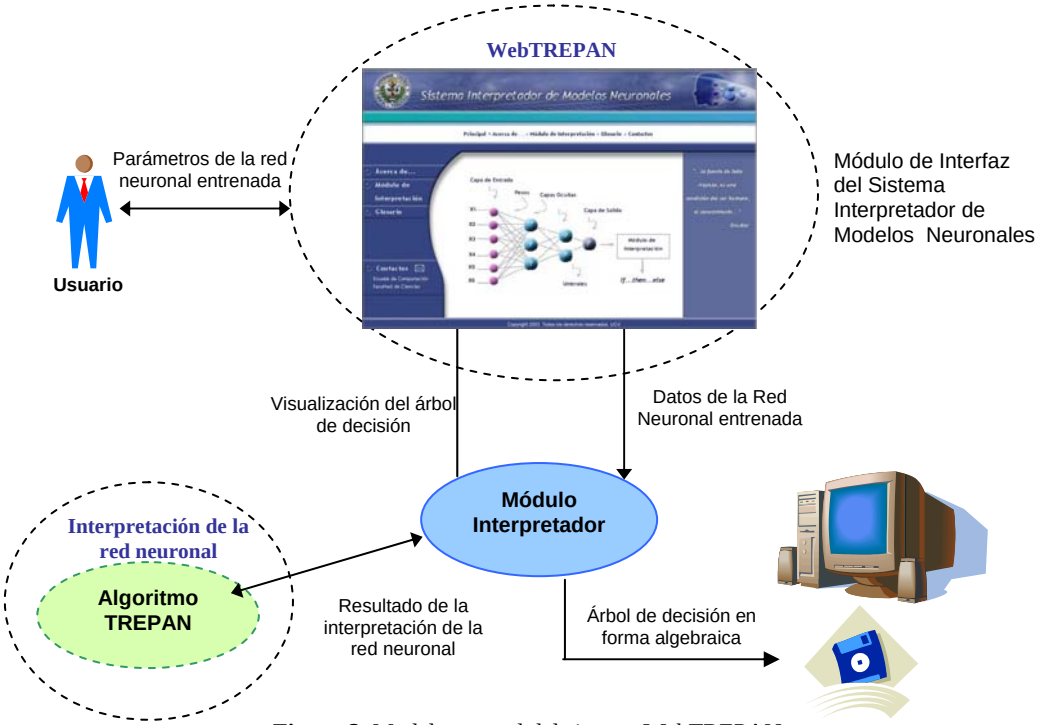


Figura 2. Modelo general del sistema WebTREPAN

La herramienta computacional está basada en Web, y mediante una interfaz humano-computador en forma de "Wizard" se pueden introducir los datos relacionados a la red neuronal entrenada que el usuario desea interpretar (tales como topología de la red, matrices de pesos y umbrales, entre otros). El "Wizard" forma parte de un módulo interpretador dentro del sistema, donde una vez que los datos han sido proporcionados por el usuario, se procede a ejecutar el método de extracción de reglas TREPAN. Posteriormente, se procede a mostrarle los resultados del proceso de interpretación al usuario a través de la interfaz del sistema; estos resultados también podrán ser almacenados en el disco duro de la PC o bien en una unidad de almacenamiento portable.

Además del módulo de interpretación, el sistema también cuenta con otras funcionalidades, como una sección que describe la justificación y los objetivos planteados, una ayuda que guía al usuario en el proceso de interpretación y de las consideraciones que debe tener en cuenta para que éste sea satisfactorio, así como una sección que suministra información del estado del proceso.

4.1 Desarrollo del Módulo de Interfaz

Para el análisis y diseño del sistema se utilizó la metodología OOSE extendida junto con el Lenguaje de Modelado Unificado UML [11],[20]. A partir del modelo de casos de uso y del modelo objeto del dominio se desarrolló un prototipo de la interfaz de usuario, que posteriormente fue refinado hasta lograr obtener la interfaz definitiva de WebTREPAN. En particular, para el diseño y desarrollo de la interfaz del sitio Web, se empleó la herramienta Macromedia Dreamweaver MX, por ser muy fácil y práctica de usar. Como programas para la edición de imágenes, se usaron las herramientas Adobe Photoshop 6.0 y 7.0; así como Macromedia Flash 5.0 y SWISH v.2 ESP en los efectos Flash que se diseñaron. A continuación, se describe el modelo de casos de uso del sistema.

4.1.1 Modelo de Casos de Uso

El diagrama de casos de uso muestra los distintos requisitos funcionales que se esperan de una aplicación o sistema y cómo se relacionan con su entorno (usuarios u otras aplicaciones) [12]. En la Figura 3 se muestran los dos casos de uso principales del Sistema Interpretador de Modelos Neuronales. El actor identificado es el *Usuario del Sistema,* que se refiere al usuario que utilizará el sistema interpretador y que podrá acceder a cualquiera de las funcionalidades que éste ofrece.

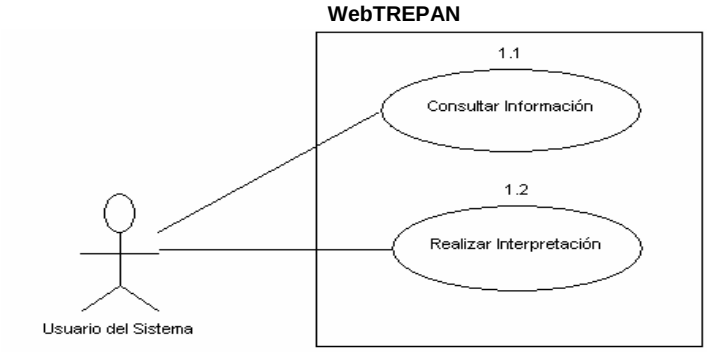


Figura 3. Interacción de los actores con los Casos de Uso del Sistema

La Figura 4 muestra el refinamiento del caso de uso *Consultar Información*. Como puede observarse, el caso de uso *Acerca de…* incluye tres funcionalidades que el usuario puede seleccionar de manera opcional para consultar información referente al sistema, como son: *Justificación*, que le brinda al usuario la posibilidad de consultar información relacionada con la justificación para el desarrollo del sistema interpretador; *Referencias*, el cual le proporciona al usuario la posibilidad de consultar información relacionada con las referencias más importantes que fueron consultadas; y *Objetivos***,** que le brinda al usuario la opción de consultar acerca del objetivo planteado en la investigación.

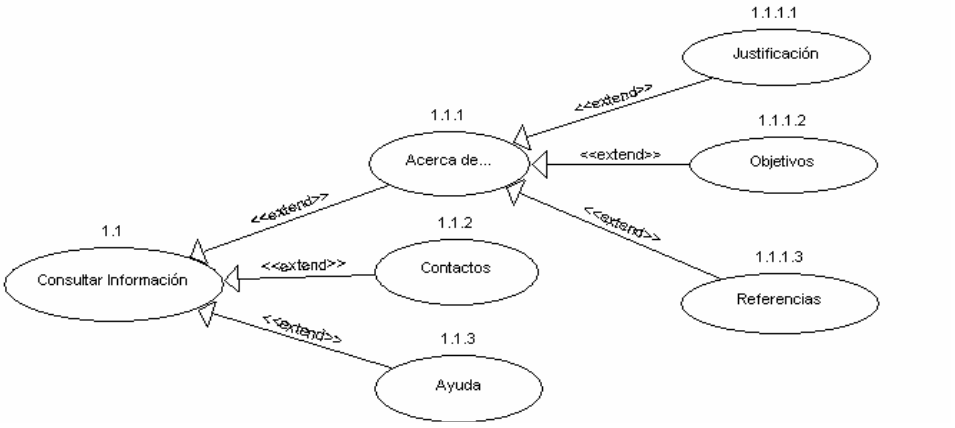


Figura 4. Refinamiento del caso de uso *Consultar información*

El caso de uso *Ayuda* es una funcionalidad donde el usuario podrá encontrar información de cómo utilizar el sistema, guiar al usuario en el proceso de interpretación y ayudarle a resolver las dudas que se le presenten para que éste sea satisfactorio. Por otro lado, el caso de uso *Contactos* le brinda información al usuario, relacionada con las personas involucradas en el desarrollo del sistema interpretador WebTREPAN.
La Figura 5 muestra el caso de uso *Realizar Interpretación,* que le permite al usuario realizar la interpretación de la red neuronal entrenada. Para realizar esta tarea, el usuario debe proporcionar todos los datos necesarios que requiere el algoritmo TREPAN. Este caso de uso incluye seis funcionalidades, cuatro de las cuales se realizan de manera obligatoria y dos que se puede realizar de manera opcional, como son:

- *Proporcionar datos de la topología*: Donde el usuario introduce los datos relacionados con la topología de la red entrenada que quiere interpretar, a saber: el número de entrada, el número de capas ocultas, el número de neuronas de las capas oculta y de salida, el tipo de función de activación por capa oculta y de salida, los pesos y umbrales de cada una de las capas.
- *Proporcionar datos de entrenamiento y prueba*: Donde el usuario selecciona los datos de entrenamiento y de test que utilizó para la creación del modelo de red neuronal.
- *Proporcionar criterio de parada*: Donde el usuario introduce de manera opcional, el número de nodos internos que desea tenga el árbol de decisión asociado a la interpretación de la red (criterio de parada global).
- *Verificar información proporcionada*: El usuario decide si modifica o no, algunos de los parámetros que introdujo como entrada al proceso de interpretación de la red.
- *Generar el árbol de decisión*: Una vez que se han proporcionados todos los datos necesarios, el usuario da inicio al proceso de interpretación de la red. Este caso de uso incluye el caso de uso que ejecuta el método de extracción TREPAN para la interpretación de la red.

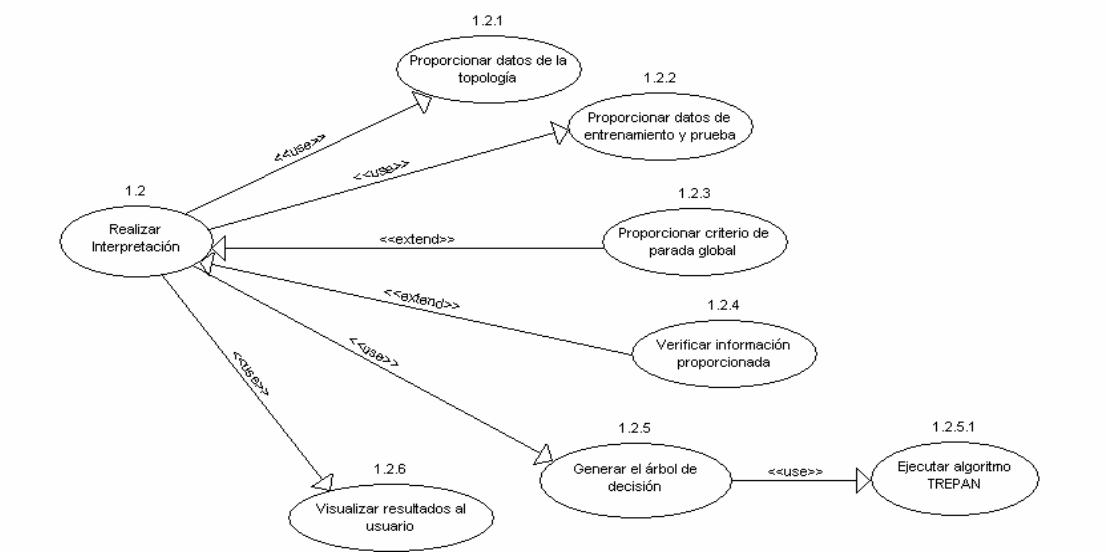


Figura 5. Refinamiento del caso de uso *Realizar interpretación*

- *Visualizar resultados al usuario*: Finalizado el proceso de interpretación, el sistema muestra los resultados obtenidos al usuario.

Para llevar a cabo la interpretación, la interfaz guiada ("Wizard"), le indica al usuario cuáles son los parámetros que éste debe introducir. Si el usuario decide realizar una nueva interpretación debe dirigirse primero a la página de inicio y desde allí acceder al módulo de interpretación, para proporcionar nuevos datos en el "Wizard".
Por otra parte, el sistema ofrece ayuda al usuario en forma de *Notas* relativas a la acción que efectúa dentro del proceso de interpretación, para orientarlo a no cometer errores o para que pueda corregirlos. Éstas se encuentran situadas en cada una de las páginas del "Wizard" en el módulo de interpretación. En caso de necesitar un entrenamiento más específico, el usuario puede consultar la ayuda global del sistema, la cual le facilita el aprendizaje del mismo. A continuación, se presentan algunas páginas que conforman la interfaz de WebTREPAN.
En la Figura 6 se muestra la página principal. En esta se puede observar el área donde se encuentran ubicados los botones que permiten acceder a cada funcionalidad del sistema y el área de presentación, donde se mostrará el contenido del "Wizard" dentro del módulo de interpretación. También hay un área destinadas a las *Notas* con información que debe conocer el usuario para realizar el proceso de interpretación. Estas notas están ubicadas a lo largo del recorrido por el módulo de interpretación y son presentadas en forma de enlaces específicos a la Ayuda del sistema donde el usuario podrá encontrar la información correspondiente.

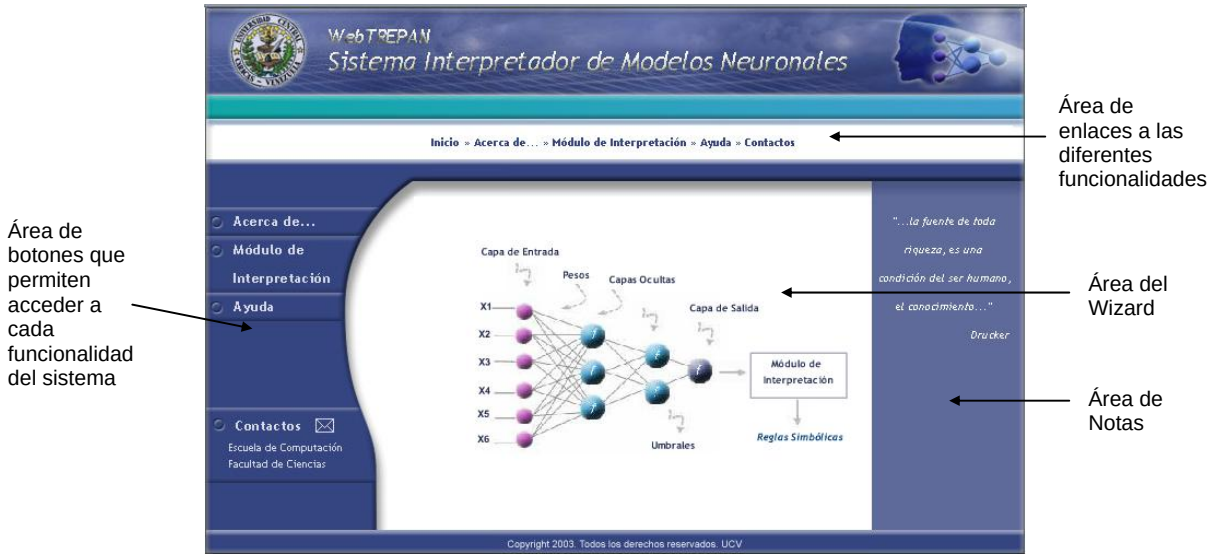


Figura 6. Página principal de WebTREPAN

La Figura 7 muestra la página principal del módulo de interpretación. Se observa el área donde se encuentran ubicadas las *Notas* a tener en cuenta para realizar el proceso de interpretación. A partir de esta página, el usuario comienza su recorrido por el "Wizard" para la interpretación de la red neuronal entrenada, según los datos que éste proporcione. Como ejemplo, en la Figura 8 se presenta la página correspondiente a la información de la capa de entrada.

Figura 7. Página principal del Módulo de Interpretación

En esta página del "Wizard", el usuario especifica el número de neuronas de la capa de entrada, selecciona la ruta del archivo de etiquetas de las variables de entrada (en caso de que haya seleccionado la opción **Si**) y escribe el número de capas ocultas que posee la red. En el área derecha de la página pueden observarse las *Notas* correspondientes, con un enlace hacia la *Ayuda* donde el usuario encontrará información sobre cómo escribir los datos de la capa de entrada y cómo escribir el archivo de nombre de las variables de entrada.

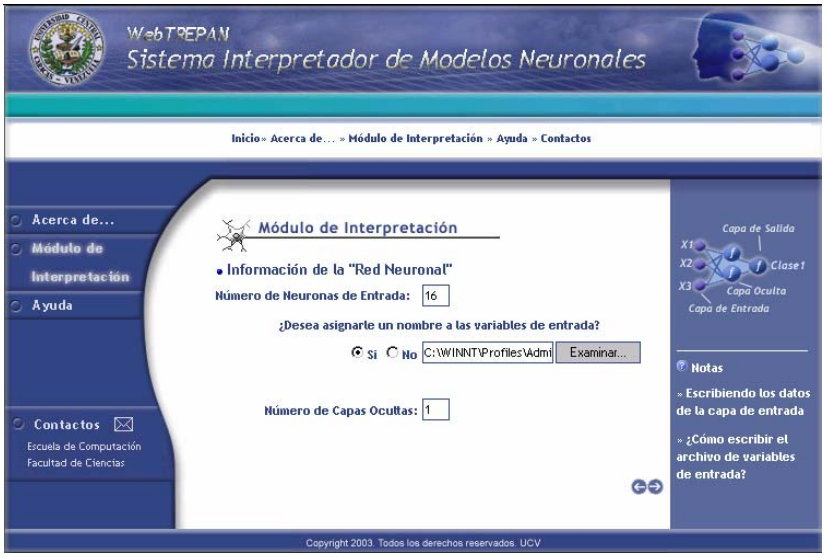


Figura 8. Módulo de Interpretación. Primera página de datos de la red neuronal

4.2 Desarrollo del Módulo de Interpretación

El proceso de desarrollo del módulo de interpretación se dividió en tres partes: la estructuración y formato de los archivos de entrada a la herramienta, la implantación de cada una de las rutinas del algoritmo TREPAN y por último, la visualización del árbol generado.

4.2.1 Estructura y Formato de los Archivos de Entrada

Los datos relacionados a la red neuronal entrenada que serán la entrada al sistema, vienen dados en 5 archivos, donde el usuario debe almacenar la siguiente información: los *datos de entrenamiento*, los *datos de test,* los *pesos y umbrales* de las capas ocultas y de salida, un archivo donde el usuario colocará los *nombres de las variables de*

entrada y otro archivo de clases donde colocará los *nombres de las variables de salida.* Esto último, sólo en el caso de que desee asociarle un nombre a cada variable, por ejemplo: Variable1 = *Temperatura,* Variable2 = *Humedad,* etc. Estos archivos podrán ser elaborados por el usuario, en un editor de texto como *Bloc de Notas* y podrán guardarse con cualquier nombre más la extensión ***.txt***.

4.2.2.Implantación del Algoritmo de Extracción de Reglas TREPAN

La implementación del algoritmo requirió un análisis previo de su estructura para lograr su descomposición en una serie de rutinas específicas. Particularmente, para la construcción del oráculo se diseñó una rutina que simula la red neuronal, lo cual requirió la implantación de las funciones de activación que podrían ser utilizadas en las capas ocultas y de salida de la misma. En particular, fueron implantadas las siguientes ocho funciones: *Escalón, Escalón Bipolar, Gaussiana, Lineal, Lineal Saturada, Lineal Saturada Bipolar, Sigmoide y Tangente Hiperbólica* [10].

Como se describió anteriormente, en la generación de nuevas instancias se debe tomar en cuenta que éstas deben seguir la distribución de los datos en el dominio del problema. Para calcular el modelo de distribución local en un nodo, se implantaron las rutinas para modelar individualmente cada variable de entrada según sea su tipo, discreta o continua.

Para una variable discreta, la distribución viene dada por los valores que puede tomar la variable, y por cada posible valor se indica la frecuencia relativa con que éste ocurre en el conjunto de datos de entrenamiento que llegan al nodo [9]. Para la generación de los números aleatorios que siguieran la distribución discreta, se aplicó el método de la transformada inversa [19] para valores discretos, donde se especifica que si la variable aleatoria **X** tiene una FDA F(x), entonces la variable u = F(x) está distribuida uniformemente entre 0 y 1. Por lo tanto, **X** se puede obtener generando números uniformes y calculando $x = F^{-1}(u)$.

Para una variable continua, se modeló la función de densidad a través del método de estimación de *densidad del núcleo ("kernel") gaussiano* [6],[14], cuya función viene dada por la expresión:

$$f(x) = \frac{1}{m}\sum_{j}^{m}\frac{1}{\sqrt{2\pi\theta}}e^{-\left(\frac{x-u_j}{2\theta}\right)^2} \qquad (2)$$

Donde m representa el número de ejemplos de entrenamiento usados en la estimación, μ_j representa el valor de la variable para el j-ésimo ejemplo y θ representa la amplitud del *kernel gaussiano*, que TREPAN fija al valor de $1/\sqrt{m}$. Para la generación de números aleatorios que siguen una distribución normal, fueron aplicados el método polar, que permite generar variables aleatorias normales estandarizadas e independientes [19]; y el método de la transformada inversa que permite transformar al número aleatorio normal generado con media μ=0 y varianza θ=1, y llevarlo al espacio original con $\mu=\mu_j$ y $\theta=1/\sqrt{m}$ [18].

Otras rutinas implantadas tienen que ver con la expansión del árbol a través de una *Cola de Prioridades*, la construcción y selección de los *test de partición* y los *criterios de parada*, entre otras.

4.2.3. Resultados de la Aplicación

La visualización del árbol generado se realizó utilizando una representación algebraica en forma de tripleta, donde para los nodos internos el primer elemento representa el nodo padre, el segundo elemento el hijo izquierdo y el tercer elemento el hijo derecho. En el sistema, se indican los nodos con letras del abecedario y se muestra una leyenda que especifica para cada letra (nodo) el test de partición asociado (si es un nodo interno) o la etiqueta de clase (para un nodo hoja); de esta forma, el usuario puede inferir el árbol obtenido. En la Figura 9 se muestra la página de resultados del Módulo Interpretador donde se observa un ejemplo de esta representación. Esta página también incluye un enlace a la *Ayuda* para asistir al usuario en la traslación de una forma a otra.

4.3 Aspectos Técnicos de WebTREPAN

El sistema se desarrolló para plataformas PC, bajo Sistema Operativo Windows 9x/2000/ME/XP/NT y en ambiente Web, específicamente para el cliente. Los requerimientos mínimos para el uso de la aplicación son los siguientes: Internet Explorer 5.0 o superior, Netscape 7.0 con soporte para la Máquina Virtual de Java (J.V.M), PC Pentium III con 256 MB de memoria RAM, Java 2 SDK, Standard Edition Versión 1.4.1, y Servidor Web Resin 2.0 o superior con soporte para Servlets.

Por otra parte, para la implantación del sistema se utilizó la tecnología JAVA; en particular, se utilizó la herramienta JCreator Pro Release V2.50 build 9 para la programación, Java Server Pages (JSP) para generar contenido Web de una manera dinámica, y JavaScript como lenguaje para la creación de los scripts. Para el manejo de los archivos de texto, como la lectura desde el cliente y la transferencia hacia el servidor para su procesamiento, se utilizó un Java Bean. También se usaron Servlets que son componentes Web, definidos como clases de Java compiladas que pueden ser cargadas de manera dinámica y ejecutadas en un Servidor Web que soporte Servlets.

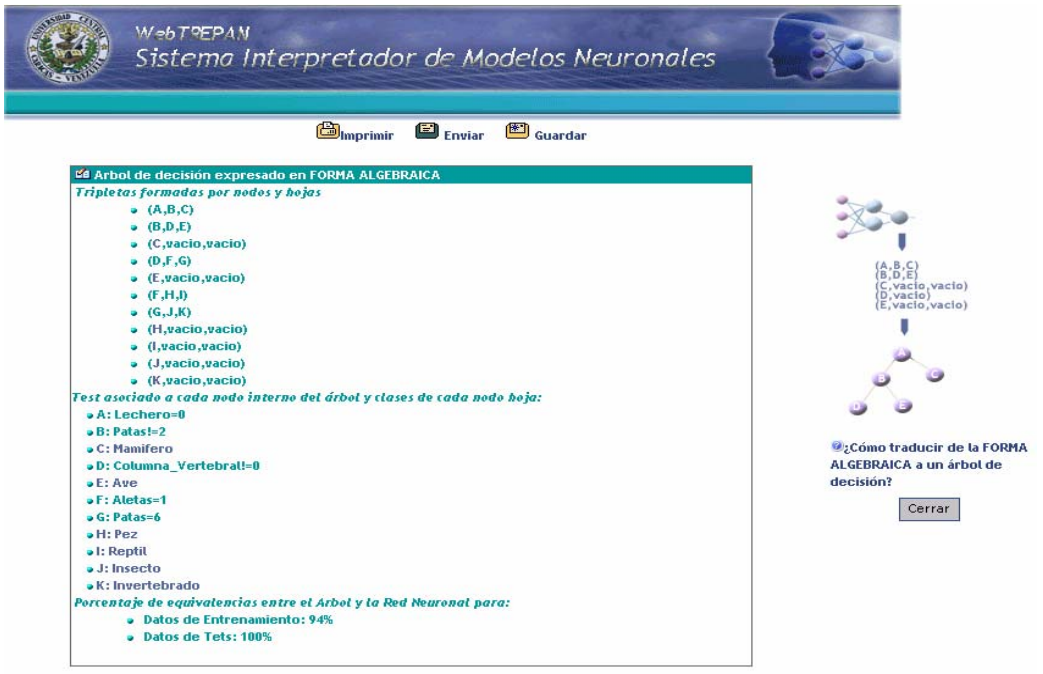


Figura 9. Módulo de Interpretación. Página de resultados

5. EXPERIMENTOS

Los experimentos que se presentan a continuación, ilustran la aplicación del sistema WebTREPAN a redes neuronales entrenadas sobre seis Bases de Datos del Repositorio UCI [5], el cual representa un estándar dentro de la comunidad del aprendizaje automático. En estos experimentos se evaluó la *fidelidad* de los árboles de decisión extraídos con respecto a la red neuronal desde la cual se realizó la extracción; esta medida viene dada por la siguiente expresión:

$$Porcentaje\ de\ Fidelidad = \frac{N°\ de\ acuerdos}{N°\ total\ de\ datos} x100 \quad (3)$$

Las características de las bases de datos utilizadas se muestran en la Tabla 1. Sobre éstas se entrenaron redes neuronales utilizando diferentes topologías y funciones de activación para evaluar la funcionalidad del sistema. Como ejemplo, en la Figura 10 se muestra el árbol obtenido (en forma algebraica y gráfica) a partir de una red neuronal entrenada con la base de datos IRIS. En esta base de datos se busca clasificar en tres clases, las cuales indican el tipo de planta IRIS (Iris-setosa, Iris-virginica e Iris-versicolor), a partir de 4 características, a saber: longitud y ancho del pétalo, y longitud y ancho del sépalo.

Tabla 1. Bases de datos y sus características

Cód.	Base de datos	No. de datos	No. de atributos	Tipo de atributos	No. de clases
1	IRIS	150	4	Numéricos (continuos)	3
2	WISCONSIN	699	9	Categóricos	2
3	WINE	178	13	Numéricos (continuos)	3
5	New-THYROID	215	5	Numéricos (continuos)	3
11	ZOO	101	16	Categóricos	7
12	HEART	270	13	Numéricos, categóricos y binarios	2

La Tabla 2 presenta los porcentajes de equivalencia de los árboles generados por WebTREPAB, con respecto a los datos entrenamiento y de test, para cada conjunto de datos.

Tabla 2. Porcentajes de equivalencia de los árboles de decisión obtenidos para las bases de datos

DATOS	IRIS	HEART	NEW THYROIDE	WISCONSIN	ZOO	WINE
Entrenamiento	94%	92%	86%	98%	94%	98%
Test	97%	93%	95%	94%	100%	94%

Árbol en forma algebraica

(A, B, C)

(B, D, E)

(E, F, C)

Leyenda:

A: 1 de 2 { Longitud del Pétalo > 4.13, Ancho sepal <= 2.96 }
B: Longitud del Pétalo > 4.77
C: Iris Setosa
D: Iris Virginica
E: Longitud del Pétalo >= 4.36
F: Iris Versicolor

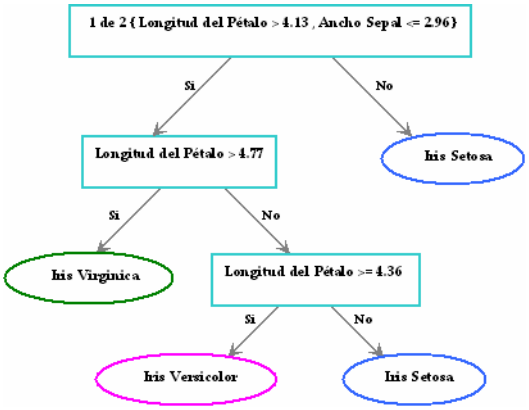


Figura 10. Árbol de decisión generado por WebTREPAN a partir de una red neuronal entrenada con la base de datos IRIS. La topología de la red es: una capa oculta con 3 neuronas, una capa de salida con 3 neuronas y función de activación sigmoide para ambas capas

6. CONCLUSIONES Y TRABAJOS FUTUROS

El Sistema Interpretador de Modelos Neuronales WebTREPAN permite extraer, de una manera automática, una representación simbólica aproximada en forma de una árbol de decisión, del conocimiento que ha capturado una red neuronal durante su aprendizaje. De esta forma, esta herramienta puede ser utilizada para facilitar la comprensión y el análisis del problema que está bajo investigación y de su solución, así como servir de apoyo en procesos de toma de decisiones por parte de expertos. Además, WebTREPAN puede complementar otras sistemas utilizados en la minería de datos.

WebTREPAN cuenta con una interfaz basada en Web sencilla y útil por medio de la cual el usuario introduce los datos de la red neuronal entrenada que desea interpretar, se le muestra al usuario el árbol de decisión en forma algebraica, y se le dan las especificaciones para que lo entienda fácilmente. Asimismo, se le brindan las consideraciones necesarias durante todo en el proceso de interpretación, lo cual reduce la posibilidad de que el usuario cometa errores. Sin embargo, se podría mejorar la visualización de los resultados de manera gráfica y dinámica, para que el usuario no tenga que realizar la traducción de la forma algebraica al árbol de decisión propiamente.

El sistema desarrollado puede ser ampliado siguiendo dos vías: una de ellas, incorporando otros paradigmas de aprendizaje considerados del tipo *caja negra*, como es el caso de las máquinas de soporte vectorial (SVM). Esto es factible ya que TREPAN es un método de extracción global; bastaría entonces con construir y añadir los Oráculos respectivos. La otra vía está relacionada con la incorporación de otros métodos de extracción de reglas para redes neuronales que produzcan otra forma de representación, como reglas de producción o autómatas de estado finito, entre otras

Referencias

[1] Andrews R., Diederich J. y Tickle A. (1995). A survey and Critique of Techniques for Extracting Rules from Trained Artificial Neural Networks. *Knowledge-Based Systems*. 8(6):373-389.

[2] Bengio, Y., Buhmann, J., Embrechts, M. y Zurada, J. (2000). Introduction to the Special Issue on Neural Networks for Data Mining and Knowledge Discovery. *IEEE Transactions on Neural Networks*. 11(3):545-549.

[3] Berthold, M. y Hand, D. (1999). *Intelligent Data Analysis*. *An Introduction*. Springer-Verlag.

[4] Bishop, C. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press.

[5] Blake, C.L. y Merz, C.J. (1998). UCI Repository of Machine Learning Data-Bases. University of California, Irvine. Dept. of Information and Computer Science. (http://www.ics.uci.edu/~mlearn/MLRepository.html).

[6] Canavos, G. (1988). *Probabilidad y Estadística: Aplicaciones y Métodos*. McGraw-Hill.

[7] Craven M. y Shavlik J. (1996). Extracting Tree-structured representations of trained networks. *Advances in Neural Information Processing Systems*. 8:24-30.

[8] Craven M. y Shavlik J. (1997). Using Neural Networks for Data Mining. *Future Generation Computer Systems*. 13:211-229.

[9] Craven, M. (1996). *Extracting Comprehensible Models from Trained Neural Networks.* PhD Thesis, Department of Computer Sciences, University of Wisconsin, Madison. USA.

[10] Haykin, S (1994). *Neural Networks. A Comprehensive Foundations*. Macmillan College Publishing Company.

[11] Jacobson, I., Booch, G. y Rumbaugh, J. (2000). *El Proceso Unificado de Desarrollo de Software*. Addison Wesley y Pearson Educación.
[12] Jacobson, I. (1992). *Object Oriented Software Engineering: A Use Case Driven Approach*. Addison Wesley.
[13] Kecman, V. (2001). *Learning and Soft Computing. Support Vector Machines, Neural Networks and Fuzzy Logic Models*. MIT Press.
[14] Mendenhall, W., Scheaffer, R. y Wackerly, D. (1986). *Estadística Matemática con Aplicaciones*. 3ra. Edición. Grupo Editorial Iberoamérica.
[15] Mitchell, T. (1997). *Machine Learning*. McGraw-Hill.
[16] Mitra, S., Pal, S. K. y Mitra, P. (2002). Data Mining in Soft Computing Framework: A survey. *IEEE Transactions on Neural Networks*. 13 (1):3-14.
[17] Negnevitsky, M. (2002). *Artificial Intelligence. A Guide to Intelligent Systems*. Addison-Wesley.
[18] Press, W., Teukolsky, S., Vetterling, W. y Flannery, B.(1992). *Numerical Recipes in C*. 2nd Edition. Cambridge University Press.
[19] Ross Sheldon, M. (1990). *A Course in Simulation.* Macmillan Publishing Company. New York.
[20] Rumbaugh, J. (1996). *Modelado y diseño orientados a objetos: Metodología OMT.* Prentice Hall.
[21] Setiono, R., Leow, W. y Zurada, J. (2002). Extraction Rules from Artificial Neural Networks for Nonlinear Regression. *IEEE Transactions on Neural Networks*. 13(3):564-577.
[22] Tickle A., Andrews R., Mostefa G. y Diederich J. (1998). The Truth will come to light: Directions and Challenges in Extracting the Knowledge Embedded within Trained Artificial Neural Networks. *IEEE Transactions on Neural Networks*. 9(6): 1057-1068.
[23] Tickle, A., Maire, F., Bologna, G., Andrews, R.; Diederich, J.: Lessons from Past, Current Issues, and Future Research Directions in Extracting the Knowledge Embedded Artificial Neural Networks. En: Wermter, S. y Sun, R. (Eds): *Hybrid Neural System*s. Springer-Verlag.
[24] Witten, I. y Frank, E. (2000). *Data Mining. Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann Publishers.

Similitud Semántica: Comparación y Crítica a los Modelos Actuales[1]

Enrique P. Latorres
Universidad ORT del Uruguay
Montevideo, URUGUAY, 11000
latorres@adinet.com.uy

Abstract.

There are several reasons for considering semantic similarity as one of the most important problems of Information Technology today. Most tasks of knowledge evaluation and scientific research, or even human common sense knowledge are accomplished by some kind of semantic similarity matching. When talking about complex problem solving, most of its complexity is that of identifying the problem itself. This means that a conceptual understanding of the problem is necessary to match its specification to the expected "input" and conditions of the solving procedure. Future systems based on knowledge must be able to reuse knowledge from different sources and should handle aspects not considered in current models. In this document many models are analyzed together with critics to their implementations or theoretic justifications, and suggests attributes that should be included in such a new paradigm.
Keywords: Semantic Similarity, Agent Conceptualization, Artificial Reasoning, Restrictions, Semantic Reuse, Knowledge Integration.

Resumen.

Hay varias razones para considerar el problema de Similitud Semántica como uno de los más importantes para la Tecnología de la Información. La mayor parte de las tareas de evaluación de conocimiento e investigación científica, o aún la aplicación de conocimiento de sentido común humano, son desarrolladas mediante algún tipo de mapeo de similitud semántica. Cuando hablamos de resolución de problemas complejos la mayor parte de la complejidad es la de identificar el problema. Esto significa que es necesaria una comprensión conceptual del problema para vincular su especificación con los parámetros y condiciones esperados para un mecanismo de resolución de problemas. En este documento se analizan muchos de los modelos utilizados actualmente junto con las críticas a sus implementaciones y justificación teórica, y sugieren atributos que este nuevo paradigma debería incluir.
Palabras claves: Similitud Semántica, Conceptualización por agentes, Razonamiento Artificial, Reutilización Semántica, Integración de Conocimiento.

[1] Este trabajo está parcialmente financiado por el Programa de Desarrollo Tecnológico (PDT) contrato S/C/BE/06/04. http://www.pdt.gub.uy.

1 Introducción

Hay varias razones para considerar el problema de Similitud Semántica como uno de los más importantes para la Tecnología de la Información. La mayor parte de las tareas de evaluación de conocimiento e investigación científica, o aún la aplicación de conocimiento de sentido común humano son desarrolladas mediante algún tipo de mapeo de similitud semántica [1][2], también conocido como "conceptual matching". Cuando hablamos de resolución de problemas complejos, la mayor parte de la complejidad es la de identificar el problema. Esto significa que se necesita una comprensión conceptual del problema para vincularlo a la mejor forma de solucionarlo. Más directamente, para resolver un problema debemos obtener la mejor correlación entre la especificación del problema y los parámetros y condiciones esperadas para un mecanismo de resolución de problemas.

Uno de los problemas más complejos es el de reutilización de conocimiento o software [3], sobre lo que se ha trabajado mucho en variadas áreas. La idea de reutilizar el esfuerzo realizado en el pasado para disminuir los costos de los desarrollos futuros, ha estado en el tapete desde siempre. La puesta en práctica de esta idea es tan variada como actividades y paradigmas hay en el desarrollo de software. Se ha intentado reutilizar, con variada fortuna, subrutinas -en bibliotecas-, objetos, componentes, programas. Pero no solo el código se reutiliza, también el diseño (e.g. Los patrones o *patterns*) y también los requerimientos (e.g. La especificación de seguridad informática en el *Common Criteria*).

Las ideas de "qué es lo que debe reutilizarse" entre distintos proyectos de software es aún más un arte que una ciencia. Sin embargo algunos patrones parecen mantenerse:

a) Si la reutilización es factible, la ganancia de rentabilidad crece con el nivel de abstracción del objeto usado.
b) La factibilidad de la reutilización depende del "lenguaje" en el que se expresa lo que se espera reusar.
c) La reutilización aparece tanto en cosas de audiencia amplia (e.g. en cosas comunes a muchos sistemas) como en cosas de dominio restringido (e.g. los programas de un cierto dominio de negocios).
d) La reutilización depende de otras cosas que aún no sabemos prever.

En general cualquier mecanismo de reutilización implica un cierto nivel de clasificación y de similitud semántica a efectos de agrupar conceptos y asociar problemas a mecanismos de solución [4]. Esto nos lleva a buscar soluciones al problema de similitud semántica. Similitud Semántica es encontrar qué tan similar es un concepto, o una pieza de software, o de información, o servicio, o proceso a alguna otra, a efectos de poder integrar, interoperar y/o reutilizar, economizando en esfuerzo y recursos.

Estos procedimientos de reutilización y correlación problema-solución están basados principalmente en actividades y decisiones humanas. Por lo tanto son caras, lentas y con propensión a errores, y hace que el esfuerzo necesario para reutilizar no sea tan atractivo como debiera, debido al costo de implementar los procedimientos que garanticen la calidad de las unidades reutilizadas.

El problema de similitud semántica debiera ser uno de los principales para la ingeniería del conocimiento. Una solución adecuada para este problema podría permitir respuestas a muchos otros. Esto tiene que ver con resolución inteligente y automática de problemas, como ser integración, interoperación y reutilización automática de conocimiento y bases de datos, traducción automática de especificaciones formales en código, ingeniería reversa automática, reconocimiento de patrones, identificación y recuperación de información automática, razonamiento basado en casos, aplicabilidad de problema/solución, y muchos otros que por ahora necesitan supervisión humana. Es seguro que soluciones a este problema podrían cambiar la efectividad y la aplicabilidad de los sistemas de inteligencia artificial en el futuro[5].

Este trabajo es parte de una investigación en la búsqueda de modelos para resolver la similitud semántica en forma computacional y automática. Muchos de los conceptos aquí vertidos son un resumen de los resultados de esta investigación de tesis doctoral.

2 Similitud semántica

La investigación en similitud semántica tiene una larga historia pero solo recientemente fue estudiada en forma más o menos amplia. Los investigadores que trabajan en similitud cubren varias disciplinas, desde la psicología cognitiva, pasando por la neurofisiología, la filosofía y la lingüística, hasta la inteligencia artificial por nombrar algunas. Asimismo, las aproximaciones de las diferentes disciplinas tienden a influenciarse unas a otras por medio de referencias cruzadas en publicaciones e investigación. Por lo que diferentes ideas tienden a ser re-expresadas y aparecen en diferentes formas y relevancia.

Un iniciador de los modelos de similitud semántica fue Russell [6], con su teoría de clases y similitud. Desde el comienzo su intención de hacer un modelo que pudiera considerar todos los aspectos del mundo real fue en vano, al

ser sensible a varias paradojas. Una de esta paradojas, el problema de "la clase de clases", lo compartió con Frege para encontrar una solución "Ninguna clase puede ser una extensión de una super-clase que cae debajo de la primera", resolviendo el problema, pero haciendo que la solución sea mucho más restrictiva, y al final menos capaz de representar la realidad. Russel, trabajando por su cuenta, desarrolló la teoría de proposiciones, pero de todas maneras volvía a caer nuevamente en ciertas situaciones paradójicas. De todas formas, estos modelos afectaron la forma de percibir los problemas de la realidad entre los futuros filósofos, los lógicos, y desde ellos hasta los teóricos de la computación de hoy.

Price [7] agrega a este modelo el concepto de atributos que pueden compartir similitudes, y separó la visión ideal de especificaciones de la realidad con la visión parcial percibida por los humanos. Este autor concluye que la determinación de pertenencia a una familia por la experiencia, muchas veces es imposible, pero es alcanzada a través de la similitud de los atributos percibidos. Más aún, similitud puede no estar relacionada a una jerarquía de clases como ser el caso de una estatua y la persona que esta representa.

Osgood [8] analizó el significado de las palabras experimentando con la interpretación humana desde un punto de vista psicolinguísta. Este tipo de evaluación, basado en el análisis de humanos realizando evaluación de sinónimos, también es realizado por Rubinstein y Goodenough [9]. Este trabajo fué validado luego por Miller y Charles [10]. Los resultados de estos experimentos son utilizados en varios estudios recientes para probar y validar estadísticamente varios algoritmos basados en computadora a efectos de determinar la similitud semántica entre conceptos expresados por palabras. Se fundamentan en la idea de que el juicio humano se considera correcto por definición, tal como es afirmado por varios autores [24]. Estos aseveran que la similitud es lo que hacen los humanos, por lo que comparan el desempeño de los algoritmos de computadora con el desempeño medido en una muestra de humanos realizando la tarea.

Si bien varios de estos autores expresan que hay algún tipo de relación inversa entre la similitud semántica y el concepto de distancia semántica, la formula exacta es muy discutida, y varía ampliamente de un autor a otro. Algunos de los modelos implementados pueden ser clasificados en pocas categorías: Taxonómicas, por Atributos, Mixtas, de Grafos de Dependencias, etc. Varias particularidades están presentes en varios modelos, por lo que es difícil hacer una clara separación entre ellos. La clasificación fue realizada principalmente basada en el estereotipo y sus atributos más relevantes.

Nótese que estos modelos de similitud devuelven valores que son diferentes del caso bivaluado: *similar={verdadero, falso}*. En muchos casos se puede asociar un concepto *fuzzy* de similitud.

2.1 Modelos Taxonómicos

Muchos investigadores evalúan la similitud semántica de acuerdo a una jerarquía de clases preestablecida (taxonomía), algún tipo de ontología que especifica las relaciones entre conceptos. Las implementaciones de similitud semántica en el enfoque taxonómico involucran en general la cuenta de bordes o vértices (relaciones) entre dos conceptos con o sin algún tipo de ponderación en los enlaces relevantes y en otros atributos.

2.1.1 Cuenta de vértices (Edge count), Densidad Normalizada (Normalized Density), Densidad de Distancia (Distance Density)

Varios investigadores han basado sus modelos en la distancia entre dos conceptos mediante la cuenta de vértices topológicos que son recorridos en una taxonomía o red semántica, desde un concepto hasta otro, sobre la base de la sugerencia de Rada [11], generalmente con alguna ponderación afectando a cada vértice. Pequeñas modificaciones a este modelo tienen variaciones significativas en efectividad. También, la taxonomía y el tipo de relaciones consideradas afecta en forma importante la efectividad del modelo, como se puede ver en el trabajo de McHale [12].

Ramon y Van Laer [13] definieron una medida de distancia entre átomos de primer orden que tiene todas las propiedades de una métrica. En forma similar a otros modelos basados en taxonomías, la distancia es calculada al comparar la distancia de vértices a una superclase común con ponderaciones. Otro modelo es el de Distance-Density de Krumhansl [14]. En este se asume que en regiones densas de estímulo, o sea clases que tienen muchas subclases, se deben hacer discriminaciones más detalladas que en zonas menos densas.

2.1.2 Contenido Probabilístico o de Información

Resnik [15] sugirió que cada cuenta de vértices sea corregida por una ponderación probabilística estimada empíricamente. Este autor consideró que la información contenida en un mensaje se puede medir como el negativo del logaritmo de la probabilidad de que ocurra el mensaje, siguiendo la propuesta de Shanon [16], aunque no lo citó explícitamente. El contenido de información es calculado a partir de un objeto al analizar la probabilidad de pertenecer a una cierta super-clase. Esto puede ser visto también como una adaptación de conocimiento estadístico a las dependencias de las relaciones. Asímismo se puede considerar que hay un cierto nivel de razonamiento

(estadístico en este caso) en la evaluación de similitud. El modelo fue comparado con los resultados de Miller y Charles, los cuales tienen una correlación del 79%.

Resnik mostró algunos aspectos importantes sobre distancia semántica. Similitud Semántica es un caso especial de relación semántica, como en el caso de *car-gasoline* (automóvil-gasolina) con relación a *car-bicycle* (automóvil-bicicleta) sobre la base de Wornet[17]. Intuitivamente, los primeros dos conceptos están más relacionados pero el segundo par tiene más similitud. Ante esto plantea que no alcanza con considerar solo las relaciones de hiponímia e hipernímia (subclase y superclase) sino también la intención y uso del concepto, ya que su modelo se basa en la sugerencia de Rada [11]. El trabajo de Resnik es similar al de Leacock y Chodorow [18]. Estos últimos consideran el largo del camino normalizado al dividir por la altura máxima de la taxonomía. Resnik mostró como las conexiones entre conceptos tienen conceptualmente e intuitivamente una distancia diferente. También Leacock y Chodorow sugieren que [18], dependiendo del caso, cualquiera de ambos métodos puede tener mejor resultado, por lo que la conveniencia de usar el método de contenido de información depende de la taxonomía y la cantidad de datos para superar a la cuenta de vértices normalizada.

Dekang Lin [19] definió otro modelo de similitud normalizado basado en teoría de la información. Lin asume que el método debe derivarse de ciertas suposiciones sobre propiedades de la medida de similitud. Este modelo toma de Tversky [29] que en la medida de similitud debe considerarse tanto los atributos comunes como los que los diferencian. Lin mostró como su modelo es un *framework* para desarrollar otras medidas de similitud para diferentes tipos de elementos, incluyendo valores a los que se pueden asociar aseveraciones de cualidad difusa. El experimento en similitud de palabras brinda resultados impresionantes luego de analizar un corpus extenso basado en tripletes de dependencias. Esto mostró una gran potencia para la recuperación de corpus de grandes textos. Lin comparó sus resultados con los de Miller y Charles [20], y obtuvo una mejor correlación que otros modelos, tales como los de Resnik [15], Wu y Palmer [21], además de otras variantes de similitud como ser la distancia de cadenas de caracteres de Levenshtein [22].

Jiang y Conrath [23] utilizaron un modelo similar al de Lin, pero su pequeña modificación tiene resultados importantes de mejora. Curiosamente su modelo también recuerda a la fórmula de Shanon de transmisión de mensajes en un medio con ruido.

Budanitsky y Hirst [24] mostraron como esta simple modificación incrementa notablemente la efectividad de la medida de similitud. El modelo de Jiang y Conrath [23] presenta una correlación del 85% con el experimento de Miller y Charles. Esto se aproxima bastante bien al máximo estimado de 88% en la duplicación del experimento de Miller y Charles[10] realizado por Resnik [15] y al máximo estimado de 90%. Pero todos estos modelos son comparados contra experimentos basados en un muy pequeño corpus fijo.

2.1.3 Intencional

El modelo de Hakimpour y Geppert [25] está basado en la detección de relaciones de similitud o diferencia. Estas son descriptas por relaciones de semántica intencional, proposiciones basadas en el modelo de Goh et al. [26]. Una sugerencia interesante en este trabajo es que se debería intentar trabajar con lógica multivaluada (difusa) para obtener mejores resultados y que se necesita de un mecanismo de razonamiento para el proceso de mapeo.

Otro modelo de similitud semántica es el propuesto por Rodríguez [27][28]. Se basa en contenido de información, pero agrega ponderaciones diferentes a los atributos comunes y a los atributos diferenciadores, tanto del referente como del referenciado. Está basado en los trabajos de Tversky [29]. Lo más innovador de su trabajo es la incorporación del concepto de "intención de usuario". Esto consiste en una implementación sencilla de contexto de uso, mediante un campo que tiene información clasificatoria. A pesar que la información de contexto es un simple atributo, los resultados mostraron que es una opción interesante a considerar.

2.2 Similitud de Atributos

Tversky [29] al analizar la forma en que los humanos comparan conceptos y objetos deriva que no solo los atributos iguales, sino que también los atributos diferenciadores son considerados en la evaluación. Más aún, los atributos de similitud y diferencia (*Common and Distinctive Features*) varían de acuerdo a la intención de la comparación, o con otro nombre, al contexto de significancia. Este trabajo ha influenciado a muchos otros y ha sentado bases para los modelos de similitud del futuro.

Bousquet et al. [30] mostraron un modelo usado en similitud de términos médicos. Como otras áreas del conocimiento científico con alto nivel de desarrollo de nuevos conceptos por varias personas, hay una fuerte tendencia a instanciar conceptos similares con diferentes nombres e incluso diferentes taxonomías. Este problema se repite y solo disminuye (sin desaparecer) cuando se llega a un consenso en el dominio de conocimiento, en algún futuro, y se estabilizan los cambios. Este problema es presentado por Cohen et al [31]. Constituye un problema muy importante para la construcción de ontologías y la educción de conocimiento en dominios altamente evolutivos

como ser el campo científico [32]. Otras tareas que se basan en similitud semántica sufren de los mismos problemas [33][34]. A no ser que se obtenga un modelo efectivo de integración y búsqueda de conocimiento en forma automática, va a ser difícil resolver este problema [3].

Bousquet et al. [30] sugirieron el uso de varios atributos organizados en una estructura similar a una taxonomía, y a cada diagnóstico se le asigna una combinación de conceptos. Esto es consistente con la percepción intuitiva de que problemas del mundo real tienen muchas veces causalidad compleja y que una sola taxonomía no es suficiente para analizar todas las relaciones con las que se compara un problema. El sistema dispone de una red semántica basada en las definiciones de SNOMED y se agrega una relación de causalidad. Las distancias se miden por todos los caminos entre conceptos y los arcos son pesados de acuerdo con los atributos considerados. Esto recuerda al modelo de Rips et al. [35] de ejes que describen atributos o relaciones de conceptos.

2.3 Basados en Teorías o en Conocimiento

Algunos modelos están basados en reglas proposicionales o de algún otro tipo que describen los objetos con los que se desea comparar. En general estas estrategias están incorporadas en sistemas mixtos donde combinan razonamiento con otros modelos. Ver [36][37][38] por ejemplos. Hay aún pocos trabajos basados en estos modelos en el área de IA, pero si hay gran cantidad de trabajos en el área de psicología cognitiva y psicolinguística que mostraron la relevancia en el pensamiento humano del razonamiento como parte del proceso de similitud [4]. Un modelo que parece promisorio en esta área es el de Razonamiento Basado en Contextos [39].

Tanto los modelos basados en teorías como los basados en atributos pueden ser utilizados como modelos de analogía para **ejemplares paradigmáticos o prototipos** ya sea que estos son dados o calculados, en estos casos la clase o categoría a la que se pertenece es representada por uno o más modelos estereotipo sobre los que se compara los atributos o las proposiciones de la teorías[40].

2.4 Mezcla de Taxonomia-Atributos y otros

Yamada, Inuzuka y Seki [41] crearon un modelo de similitud basado en la creencia que dados dos elementos, a mayor posibilidad de pertenecer a una clase, mayor es la similitud de ambos elementos, donde la mayor probabilidad es cuando pertenecen a la misma clase. Una métrica de similitud no existe por si misma sino que es una solución al problema en el caso considerado. Para ello se necesita un cierto punto de vista de interpretación. Los atributos comunes y diferenciadores de los objetos son medidos en relación a la cantidad de información contenida en una proposición que describe las diferencias e igualdades. Se asume que la descripción del objeto puede ser dividida en descripciones de perspectivas independientes. La similitud es entonces un promedio ponderado de las similitudes entre descripciones de los objetos de cada perspectiva. Esto recuerda mucho al modelo basado en contextos [39]. La métrica de similitud utiliza un modelo de probabilidad condicional del contenido de información. Luego se adapta este modelo a un modelo de clasificación basado en *k-Nearest Neighbor* (Instance Based Learning) y se obtiene un modelo que puede ser entrenado.

2.5 Dependencia por Grafos.

Melnik, Garcia-Molina y Rahm [42] definieron un método usado en un sistema para mapeo de esquemas de bases de datos, mediante mapeo de grafos. A partir de una función SQL2Graph se convierte la definición del esquema de base de datos a un grafo, y los grafos obtenidos son comparados por un método iterativo. Ellos se basan en la intuición básica de que elementos de diferentes grafos son similares cuando los elementos adyacentes son similares. Los cálculos se realizan sobre la base de los nombres de las etiquetas del grafo en forma recursiva hasta que los valores de las métricas y las ponderaciones se estabilizan. Si el sistema no converge se detiene luego de un número predeterminado de iteraciones.

El trabajo de Hasegawa et al. [43] mostró algunas experiencias en un motor de categorización semántica de documentos. Se utilizaron analizadores conceptuales que convierten al texto en diagramas conceptuales [67]. Luego se compararon los grafos supuestamente canonizados para identificar las similitudes.

3 Criticas

En general las actuales implementaciones de todos estos modelos incluyen alguna deficiencia en su concepción de la que deriva en un pobre desempeño o justificación teórica. Algunos de estos son descritos a continuación.

3.1 Modelos Fuera de Contexto o de Contexto Neutro

La mayoría de los modelos consideran la similitud semántica entre palabras o conceptos fuera de contexto. Estos modelos no consideran información situacional o de contexto, términos que son frases u otros símbolos conceptuales. Esta búsqueda de conceptos y categorías fuera de contexto o de lenguaje neutro es tan largo como nuestra historia y ha tenido muy poco éxito. Comenzó con los modelos aristotélicos de la realidad y que han prevalecido en el pensamiento humano por los últimos 2500 años. Paradójicamente, Aristóteles dice que él no es el

verdadero experto en el tema, y que de haber tenido los 50 dragmas para presenciar el curso de Pródicus sería capaz de decir más [44]. Más allá de que muchas de las interpretaciones de este texto suponen una cierta ironía de parte de Aristóteles, la realidad es que solo han llegado a nuestros días las ideas de Aristóteles a través de los escritos de Platón y poco o nada de las ideas de Pródicus.

En introspección vemos que la mente humana no maneja conceptos independientes del contexto [45]. Ya sea que el concepto se asocie a otros conceptos percibidos y/o memorizados [2] o a situaciones de uso, la psicología cognitiva [29][46][47] y la lingüística[48] estan entrando en el consenso de que no es posible razonar o interpretar en forma inteligente con conceptos libres de contexto.

Aún para análisis de palabras fuera de contexto, los humanos tratamos de contextualizar los términos para aplicar el conocimiento disponible, ya sea considerando la información del agente que envió el mensaje, el medio por el que fue recibido, experiencias anteriores relacionadas con este, etc. Este proceso de contextualización es una parte central del proceso de percepción y de razonamiento de similitud [49], ya que cualquier mensaje percibido debe ser conectado al conocimiento existente del agente receptor [2] a los efectos de asociarlo al conocimiento tácito que este posee. Nótese que en este caso el contexto es solamente un caso especial de conocimiento acerca del concepto, sus restricciones y aplicabilidad.

Esto plantea la necesidad de desarrollar e incorporar mecanismos de evaluación conceptual asociados a contextos específicos, dentro de los modelos de información conceptual en los sistemas de IA.

3.2 Relaciones a considerar y Taxonomía.

Muchos modelos están basados en experimentos limitados a relaciones de sub-clase y super-clase, también conocidos como relaciones *is-a* o hiponímia/hipernímia. Otros han incorporado relaciones de holonímia/meronímia o *has-part*. Pero algunos experimentos con muchas más relaciones y taxonomías más abiertas sugieren que tomar en cuenta más tipos de relaciones es mejor, aún cuando esas relaciones no estén clasificadas ni se use tipo alguno de ponderación, más que la simple cuenta de vértices [12][50]. Por otra parte hay varias experiencias que mostraron que las visiones tradicionales de llevar la representación de conceptos a clases, modelos ejemplares o prototipos va en contra de la percepción humana, por lo que no serían modelos adecuados para representación de sistemas y requisitos complejos. Ver en [4][51] algunas revisiones que comparan contra muchos experimentos en el área de psicología cognitiva.

Stuckenschmidt [72] en sus conclusiones mostró como su sistema tuvo problemas para mapear conceptos cuando no se podían subsumir. Esto mostró o que el sistema carecía del conocimiento necesario, o que muchas representaciones no tenían suficiente estructura taxonómica para representarlos, o finalmente que usar solo información taxonómica no es suficiente para representar la similitud a efectos de resolver la integración de la información. Hakimpour y Geppert [25] mostraron conclusiones similares. Otros ejemplos sugieren también las limitaciones de las taxonomías como modelo.

Guarino y Welty [52] mostraron varios problemas de implementación de ontologías y sugieren la ventaja de reutilizar algunos modelos provenientes de la escuela ontológica de la filosofía. Las ontologías que se suponen deben traer orden y estructura a ciertos dominios de conocimiento son generalmente confusas y de pobre estructura taxonómica, muchas veces debido a un uso incorrecto de las relaciones *is-a*. Algunos modelos tratan de resolver esto separando los tipos de razonamiento, muchas veces en niveles de abstracción, o en tipos de teoría, como ser sintáctico, estructural y semántico. De esta manera separan los Tbox y los Abox en sus modelos de lógica y representación [53].

Pero como la información semántica es difícil de definir y como es representada por conceptualizaciones de especificaciones más o menos concertadas entre varios individuos, pero no universales, mediante un consenso amplio, son muy difíciles de obtener. Un ejemplo simple sobre lo difícil de la universalidad de los conceptos se puede ver al comparar entre varios diccionarios de cualquier lenguaje las definiciones de varios términos, especialmente términos utilizados con significados especiales en dominios de conocimiento específicos.

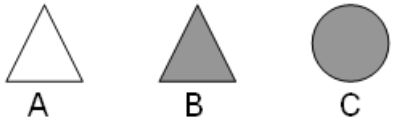
Entonces las ontologías en su interpretación usual de estructuras taxonómicas arbóreas, pueden ser usadas solamente en pequeños dominios con alcance limitado a donde se pueda obtener el consenso entre los diferentes actores. Pero algunos dominios como ser el lenguaje natural, son de por sí imposibles de ser capturados por la conceptualización simple de una taxonomía de este estilo [54]. Estos problemas de obtener consenso en las definiciones de clases se extienden al de obtención de ejemplares y de prototipos ya que en cualquiera de estos casos se debe hacer un compromiso sobre los mismos.

Por lo tanto estos niveles de representación (léxica, estructural y semántica) no debieran ser separados ya que están fuertemente interrelacionados. Los términos se clusterizan alrededor de conceptos, y la información sintáctica está

fuertemente relacionada a la interpretación semántica. El lenguaje natural captura mucha información contextual y situacional junto al concepto central. Al mismo tiempo dos frases en lenguaje natural referidas al mismo episodio no tendrán la misma representación ya que diferentes términos y especialmente *near-synonyms* tendrán diferente información situacional [54], haciendo imposible manejar todas las combinaciones en cualquier modelo computacional.

3.3 Información tácita y desigualdad triangular.

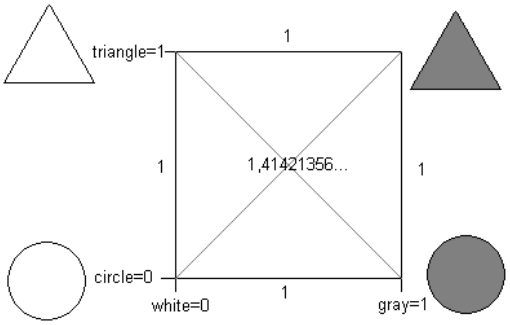
En el ejemplo de Lin [19] sobre similitud de formas geométricas, se muestra el problema de la desigualdad triangular. Considerando que la similitud es inversa a la distancia semántica, el problema entonces es el siguiente: *dist(A,C)<=dist(A,B)+dist(B,C)*. En otras palabras, si A es similar a B en una cantidad a, y B es similar a C en una cantidad b, entonces A debe ser similar a C en una cantidad c tal que c<=a+b. Y esto debería tener sentido aún para comparaciones difusas como ser {muy, mucho, un poco, etc.}. Por ejemplo dado los siguientes elementos A, B y C.



A y B tienen forma similar, pero B y C tienen similar sombreado. ¿Qué tan similares son A y C? Da la impresión que la similitud no es transitiva cuando se aplica a conjuntos disjuntos de atributos. En el ejemplo se ve que la intuición de mantener la desigualdad triangular no tiene sentido ya que lo que se quiere comparar tiene tan pocos atributos y tanta diferencia entre ellos, que el conjunto de atributos compartidos por todos los elementos es el conjunto vacío.

Para la mayor parte de los casos como este, la desigualdad triangular no tiene sentido a no ser que se identifique un conjunto de atributos relevantes para comparar conceptos dentro de un determinado contexto. Entonces el problema de la desigualdad triangular depende fuertemente del contexto de aplicación y la intención de la comparación entre conceptos, concuerda con los trabajos de Tversky en similitud humana [29].

Esta desigualdad toma sentido si se le agrega un modelo topológico o de coordinadas para comparar los atributos. Si asumimos un conjunto de coordenadas para cada atributo del concepto, donde la forma {*circle*, *triangle*} y el color {*white*, *gray*} son arreglados en ejes perpendiculares, entonces se puede derivar una medida de similitud, aún cuando la similitud intuitiva puede parecer sin sentido. Los métodos de resolución de problemas arbitrarios en humanos consisten en encontrar el contexto adecuado para la interpretación y comparación de los conceptos [48]. En el siguiente modelo, la similitud total quiere decir distancia cero, mientras que la diferencia en uno o dos atributos será de 1 o 1,41421356... (raíz de 2) respectivamente.



En este caso estamos comparando formas geométricas simples, pero los humanos como agentes cognitivos pueden derivar otros atributos y relaciones no triviales, como ser posición temporal o espacial, rugosidad de la superficie, o cualquier otra información que pueda ser derivada mediante razonamiento sobre la base de los símbolos y la situación analizada. Un agente cognitivo artificial debería poder operar en forma similar.

Una comparación por similitud es un proceso que devolverá un resultado útil si es relevante, y existe información para desambiguar el problema, de otra manera no se puede asegurar nada sobre la situación. Esto es especialmente cierto cuando los atributos similares y diferenciadores no pueden ser determinados con antelación (como en el caso de la presencia de un nuevo estímulo nunca antes percibido por el agente).

Al considerar similitud conceptual debemos incluir la información tácita ya que el token (símbolo) del concepto no explica nada acerca de su significado.

3.4 Contenido de Información Ponderada, Modelos Difusos y Probabilísticos

Como primer tema a notar es que la mayoría de estos modelos se han probado contra conjuntos muy reducidos de conceptos, en varios casos comparados contra el trabajo de Miller y Charles[10]. Entonces estos modelos son relevantes solo si lo que se compara es información fuera de contexto. Pero está implícito anteriormente, los problemas del mundo real deben ser conceptualizados en un contexto determinado por lo que estos modelos en general no parecen ser aplicables a problemas del mundo real.

Estos sistemas tienden a caer en algunos problemas recurrentes, como el identificado por Resnik [15] relacionado con el uso incorrecto del significado de una palabra. Significa que la polisemia debe ser resuelta antes de elegir el concepto correcto, de lo contrario la similitud se basaría en un error de categorización [1][55]. Esto mostró la importancia del contexto en el que las palabras son interpretadas [48], para asegurar que el concepto correcto se mapeado al término usado. Sin un método para extraer el significado correcto de una palabra, Resnik sugiere usar

todos los conceptos asociados con los dos términos a comparar de acuerdo a una medida de relevancia [56]. El sugiere la combinación de varios métodos de abducción de conocimiento y desambiguación de palabras a efectos de identificar el significado correcto de una palabra [57].

Por otra parte, los modelos basados en contenido de información consideran solamente el largo o la cantidad de conceptos sin considerar su significado. Por lo que, extender estos criterios de información conceptual es bastante irreal [58]. Lo dicho se cumple también para muchos trabajos basados en extensiones del de Resnik. Dekang Lin [19] mostró que la probabilidad debe ser calculada por adelantado significando un costo importante cuando hay que cambiar información del sistema.

Otros modelos de similitud basados en modelos probabilísticos y/o difusos usan esquemas taxonómicos o basados en ejemplos paradigmáticos o prototipos pero cuya similitud se calcula con modelos bayesianos y/o fuzzy. El primer tema es que no hay ninguna evidencia de que el razonamiento humano se base en modelos estadísticos, más bien todo lo contrario. No es el caso de modelos difusos donde hay muchos elementos del razonamiento humano. Esto de todas formas no sería ningún problema pues lo que estamos tratando de modelar es similitud para agentes computacionales. Pero lo cierto es que estos modelos tienen buenas respuestas para grandes cantidades de casos y no pueden responder bien para situaciones límite o para conjuntos de datos con los que no han sido entrenados. Las estadísticas de estos modelos dan buenas correlaciones cuando la cantidad de datos de entrenamiento es grande. Ver [36] como un ejemplo.

Todos estos modelos son en realidad el resultado de falta de conocimiento asociado a los conceptos que se están manejando. Esto se ve reflejado en el uso del máximo logaritmo de la probabilidad condicional de encontrar un elemento común que subsume a ambos conceptos. Este cálculo es realizado considerando toda la taxonomía incluyendo conceptos no relevantes.

3.5 Modelos basados en Nearest k Neighbors

Estos modelos tienen buenas aproximaciones a los resultados pero no permiten manejar cualquier tipo de excepción, como las que ocurren bastante frecuentemente en el mundo real y en el Procesamiento de Lenguaje Natural (NLP) o en la Abducción Automática de Conocimiento (AKE). Los modelos basados en *k Nearest Neighbor* asumen un conjunto fijo y conocido de atributos para la clasificación de los elementos. Esto significa que no podemos trabajar en elementos cuya cantidad y calidad de atributos depende del contexto y/o razonamiento sobre el concepto. Ver el ejemplo de Yamada et al.[41].

3.6 Criterios de Similitud Humana

Como se indicó antes, algunos autores han considerado a la evaluación humana como paradigmática y si bien no se puede reproducir, se busca que los algoritmos imiten el desempeño y los resultados de estos experimentos. Hay que diferenciar aquellos que usan los modelos para tomar ideas para sistemas computacionales con capacidad de similitud semántica, y aquellos otros que los utilizan buscando desarrollar un modelo del razonamiento y pensamiento humano.

Tenemos la experiencia de Tversky y Kahneman [59] mostrando como los humanos somos evaluadores bastante malos que utilizan tan solo unas pocas estrategias cuando se dispone de información incierta e incompleta, y el nivel de similitud expresado es relativo a la intención y a la forma en que se expresa la solicitud.

Las posiciones considerando lo opuesto son pocas y hablan de casos aislados de expertos en su tema. Uno de los ejemplos es el de la carta de Pascal enviada a Fermat [60] con relación al señor de Meré, donde se describe en esta persona un conocimiento muy preciso sobre probabilidad aprendido de la experiencia sin el conocimiento de las matemáticas.

También la evaluación humana tiene desvíos debido a sus limitados recursos cognitivos [59], por lo tanto el uso de estos modelos sin cuestionarse la aplicabilidad y que el modelo en realidad no solamente esté reproduciendo una capacidad humana sino también sus limitaciones, los hace parecer dudosos como modelos computacionales. Muchos estudios en conocimiento humano deben manejar las limitaciones fisiológicas de la mente para manejar conjuntos (chunks) de estructuras conceptuales [61][62]. Estas son limitaciones que restringen a los modelos cognitivos humanos pero que ciertamente no queremos que se propaguen a los modelos de razonamiento computarizado.

Sobre la base de los trabajos de Tversky [29] podemos ver en tareas de diferenciación o de similitud, los humanos tienden a considerar atributos relevantes a efectos de clasificar los objetos, e ignorar otros atributos no relevantes para el proceso de clasificación. Esto sugiere el uso de algún conocimiento o meta-conocimiento para manejar los problemas de similitud. La percepción de similitud puede verse afectada por la forma como se presente el problema de similitud y otros aspectos subjetivos de los cuales no se tiene la seguridad que fueran controlados en esos experimentos [63]. Sumados estos factores parece muy discutible la validez de algunos de los experimentos citados

y el uso de estos, a no ser para obtener modelos que sirven para analizar y modelizar el pensamiento humano.

Esto abre una cierta duda en varios de los modelos computacionales basados en teorías de la psicología cognitiva, sugeridos sin el suficiente análisis de validez dentro del campo de la IA. Muchos avances en la IA están relacionados a investigaciones de la psicología cognitiva, pero la aplicabilidad de estos resultados debe fundamentarse también en la aplicabilidad de las precondiciones.

Los modelos basados en contenido de información fueron algunas veces directa o indirectamente inspirados por el trabajo de Krumhansl [14]. Este modelo está relacionado a algunos de los resultados de Tversky sobre la percepción de similitud y categorización y la relevancia de los atributos de diferencia y similitud. En una taxonomía o clase muy llena, con muchos elementos para categorizar, los humanos tienden a extremar esfuerzos para diferenciar y seleccionar los elementos relevantes a efectos de acomodar los limitados recursos cognitivos humanos, y de esta manera manejar la complejidad de este conocimiento. Entonces estos modelos de densidad de información pueden ser interesantes como modelos de conocimiento humano y sus limitaciones fisiológicas, pero probablemente serán inconvenientes o irrealistas para conocimiento tratable por computadora. Por otra parte hay que notar que Corter [64] reprodujo experimentos de similitud evaluada por humanos y no pudo reproducir el efecto de Krumhansl.

4 Conclusiones

La similitud semántica no debe basarse en una cantidad limitada de relaciones. Muchas veces las implementaciones de ontologías limitan estas relaciones a unos pocos tipos incluidos en la herramienta que les da soporte. Aunque recientemente varios investigadores han estado incluyendo nuevas relaciones, muchas otras son dejadas afuera. Finalmente todo tipo de relación debe ser considerado. Esto parece dar la prerrogativa a modelos basados en teorías o conocimiento donde la flexibilidad de implementar nuevas relaciones cuando las haya parece ser más sencillo. Pero también hay carencias en teorías adecuadas que determinen qué y cómo deben ser representadas estas relaciones, a no ser por algunos estudios de relaciones de mereológicas y teoría de clases.

También se necesita alguna información de explicaciones, información causal sobre los conceptos y relaciones que permita conectar la similitud por estos conceptos, ya sea por intención o por uso, u otro conocimiento causal. Varios estudios de Quillian [65], o de Collins y Loftus [66], apoyan la teoría que la mayor parte de los procesos de razonamiento en humanos puede ser simulados por redes semánticas y técnicas de *spreading activation*. De todas formas la mayor parte de los experimentos se basaron en estructuras limitadas como ontologías taxonómicas. Esto parece indicar que una solución podría estar en variaciones de modelos de grafos conceptuales, como los de Sowa [67], resolviendo algunas de sus limitaciones. Por otro parte estos modelos ofrecen explicaciones semánticas superiores a otros modelos lógicos y según su implementación permiten manejos de teorías prosentenciales de explicación de la realidad [68]. Algunos de los problemas de los modelos actuales de representación de grafos conceptuales pueden verse en el trabajo de Theodorakis [69].

Euzenat [70] sugirió el uso de varios lenguajes para soportar varios modelos de manejo de conocimiento, razonamiento y otras acciones en el conocimiento compartido. Pero el problema de la similitud semántica necesita un lenguaje y un proceso de razonamiento especial para sí. Un posible modelo podría identificar la relevancia de un problema y determinar el subsistema o la interfase con la que comunicar el problema a resolver en un "servidor resolvente de problemas" específico, donde operen agentes algorítmicamente eficientes.

Euzenat, explícitamente indicó que no se debe diferenciar el conocimiento de fondo o de contexto con las anotaciones formales sobre los conceptos, pues todos forman la representación del concepto, y expresó la problemática de los lenguajes de representación con el compromiso entre expresividad y complejidad y la completitud de los demostradores de teoremas.

Una solución propuesta sería un lenguaje, extensible, lo suficientemente abierto para manejar nuevos formalismos (casi como un lenguaje de programación), con capacidad de manejo de patrones, en especial patrones situacionales. Este lenguaje debe tener previsto el pasaje de información y problemas a otros agentes más especializados.

En el manejo de Lenguaje Natural o en la abducción de conocimiento frases como "seguido", "casi", "en gran medida", "la mayor parte del año", "inmediatamente después", etc. pueden ser encontrados en muchos términos de problemas reales [72]. Aunque imprecisos estos valores agregan valiosa información a la definición de problemas y pueden ser mapeados a rangos de valores.

La ambigüedad de estas frases no puede ser clarificada sin una fuerte investigación y el desarrollo de consenso en los modelos actuales, requiriendo grandes cantidades de esfuerzo, cuando no imposibles tareas. Además la extensión con la que deben ser evaluadas puede variar de acuerdo a la aplicación o al dominio donde son evaluadas. Pero un nuevo modelo de similitud dependiente del contexto podría considerar las diferencias entre las frases "Pablo pasó

cerca de Juan" y "Galileo pasó cerca de Júpiter". La mayor parte de los modelos carecen de información de contexto y conocimiento del dominio de los elementos evaluados, con la excepción de Rodríguez [27], aunque en un modelo simple.

Por lo tanto, el modelo lógico de similitud debe tener bases en modelos de redes semánticas por lo que la teoría de Razonamiento Basado en Contextos al estilo de McCarthy[71][39] no sería aplicable tal como está desarrollada hoy.

Artale et al [53] sugirieron que el manejo de múltiples excepciones es un atributo fundamental para un modelo que maneje la representación y el razonamiento de similitud. Stuckenschmidt [72] sugirió que la clave para la similitud semántica es encontrar las relaciones que vinculan a las clases y objetos, y que estas relaciones tendrán diferentes modos de razonamiento y formas. Por lo tanto, no podemos limitarnos a modelos muy estructurados que limiten nuestra capacidad de razonamiento en ciertas situaciones. Entonces, el modelo debe considerar no sólo taxonomía sino también atributos, reglas y restricciones [3]. Este tipo de lenguajes tendrá la dificultad de no poder asegurar su clausura por lo que deberá estar apoyado en heurísticas y en meta-conocimiento que aseguren los procesos de razonamiento.

Cualquiera sea el modelo de representación elegido, este debe capturar información situacional, conceptual, estructural, lexica y debe estar basado en conocimiento sobre las reglas y restricciones de toda esta información. Debe, además, representar esta información en modelos de redes más complejas y flexibles que las estructuras arbóreas de muchas ontologías. Así también, debe considerar en su conocimiento no solo la información de clases, sino también las instancias dentro del modelo, como es sugerido por Rodríguez [27].

Existe la necesidad de una nueva definición de similitud y conceptos. Entonces nuestro problema de similitud semántica depende de los modelos que definamos tanto para similitud, para conceptos, dentro de una metáfora computacional que pueda manejar a todos estos junto con el conocimiento que los describe.

5 Agradecimientos

Se agradece los comentarios de los revisores anónimos aunque razones de espacio limitan el desarrollo de algunas de las sugerencias.

6 Referencias

[1] Chi, Michelene T.H.; Creativity: Shifting Across Ontological Categories Flexibly. In T.B.Ward, S-M- Smith and J. Vaid (Eds.) *Creative Thought: An investigation of conceptual structures and processes* (pp 209-234). American Psychological Association. Washington D.C.1997

[2] Gentner, D., & Markman, A.B. Structure mapping in analogy and similarity. *American Psychologist, 52*(1), 45-56. 1997.

[3] Latorres, Enrique P.; Reuso de Reglas de Negocio: Una experiencia de reuso de ontologías en un dominio restringido. Master Thesis. Universidad de la República. Facultad de Ingeniería. Diciembre 2002.

[4] Medin, D. L.. Concepts and conceptual structure. *American Psychologist*, 44(12):1469-1481, 1989.

[5] Rota, G.C., *Indiscrete Thoughts*, F. Palombi editor, Birkhäuser, Boston-Basel-Berlin. pages 57-59, 1997.

[6] Russel, B. *The Principles of Mathematics*, Ch XXVI, 1903.

[7] Price, H.H. *Hume's theory of the external world*, 1940.

[8] Osgood, C.E. The nature and measurement of meaning. Psychological bulletin, 49:197-237, 1952.

[9] Rubinstein, H; Goodenough, J.B. Contextual correlates of synonymy, *Communications of the ACM*, 8(10):627-633. 1965.

[10] Miller, G.A.; Charles, W.G. Contextual correlates of semantic similarity, Language and Cognitive Processes, 6(1):1-28. 1991.

[11] Rada, R.; Mili, H.; Bicknell, E.; Blettner, M.; Development and application of a metric on semantic nets. IEEE Transactions on Systems, Man, and Cybernetics, 19(1):17-30, February 1989.

[12] McHale, Michael. A comparison of WordNet and Roget's taxonomy for measuring semantic similarity. In Proc. Usage of WordNet in Natural Language Processing Systems, 1998. COLING-ACL '98 Workshop, University of Montreal, August 16, 1998.

[13] Ramon, J.; Van Laer, W.; Bruynooghe, M.; Distance measures between atoms.

[14] Krumhansl, C. Concerning the applicability of geometric models to similarity data: The interrelation between similarity and spatial density. *Psychological Review* 85(5):445-4 63. 1978.

[15] Resnik, P; Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of IJCAI-95*, pages 448-453, Montreal, Canada. 1995.

[16] Shanon, C.E.; A mathematical theory of communications, Bell Systems Technical Journal, Vol.27, pp.379-423 and 623-656, 1948.

[17] Miller, G. WordNet: An on-line lexical database. International Journal of lexicography, 3(4), 1990.

[18] Leacock, C.; Chodorow, M.; Filling in a sparse training space for word sense identification. Unpublished. 1994.
[19] Lin, D. An Information-Theoretic Definition of Similarity, University of Manitoba, Canada. In *Proceedings of the 15th International Conference on Machine Learning*, Madison, WI.1998
[20] Miller, G.A.; Charles, W.G. Contextual correlates of semantic similarity, Language and Cognitive Processes, 6(1):1-28. 1991.
[21] Wu, Z.; Palmer, M. Verb semantics and lexical selection. In *Proceedings of the 32nd annual Meeting of the association of computational linguistics*, pages 133-138, Las cruces, New Mexico. 1994.
[22] Levenshtein, I.V.; Binary code capable of correcting deletions, insertions, and reversals. *Cybernetics and Control Theory*, 10(8):707-710, 1966.
[23] Jiang, J.J.; Conranth, D.W. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of International Conference on research in Computational Linguistics*, Taiwan. 1997.
[24] Budanitsky, A.; Hirst, G. Semantic Distance in Wordnet: An experimental, application-oriented evaluation of five measures. 1999
[25] Hakimpour, F.; Geppert, A.; Resolving semantic heterogeneity in schema integration: An ontology based approach. In Chris Welty and Barry Smith, editors, Proceedings of International conference on Formal Ontologies in Information Systems FOIS'01. ACM Press, October 2001.
[26] Goh, Cheng Hian; Bressan, Stephane; Madnick, Stuart; Siegel, Michael. Context interchange: New features and formalisms for the intelligent integration of information. *ACM Transaction on Information Systems*, 17(3):270-290, 1999.
[27] Rodríguez, M.A. *Assessing semantic similarity among spatial entity classes*, PhD Thesis, University of Maine, May 2000
[28] Rodríguez, M.A.; Egenhofer, M. J., Putting similarity assessment into context: Matching functions with user's intended operations, University of Maine.
[29] Tversky, A. Features of similarity, *Preference, Belief and Similarity*, Selected Writings Amos Tversky, Edited by Eldar Shafir. MIT Press 2004.
[30] Bousquet, C.; Jualent, M.C.; Chatelier, G.; Degoulet, P. Using semantic distance for the efficient coding of medical concepts.American Medical Informatics Association, AMIA 2000 Annual Symposium, Los Angeles, CA, November 2000.
[31] Cohen, P.; CHaudri,V.; Pease, A.; Schrag, R.; Does Prior Knowledge facilitate the Development of Knowledge-Based Systems? Department of Computer Science, University of Massachusetts, 1999.
[32] Menzies, T.; Cost Benefits of Ontologies, Intelligence, Fall 1999, p26-32, 1999.
[33] Cleverdon, C.; Optimizing convenient online access to bibliographic databases, Information Services and Use, 4, 37-47, 1984.
[34] Ellis, D.; Furner-Hines, J.; Willett, P. On the measurement of inter-linker consistency and retrieval effectiveness in hypertext databases, Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval, Dublin, IE, 51–60, 1994
[35] Rips, L.; Shoben, J.; Smith, E. Semantic Distance and the verification of Semantic relations. *Journal of Verbal Learning and Verbal Behavior*. 12:1-20. 1973.
[36] G. Gibbon and J. Aisbett (2000) Human categorisation and its application to automatic classification. In C. Davis, T. van Gelder & R. Wales, eds., Cognitive Science in Australia, Causal Productions (electronic) Adelaide. 2000.
[37] Kalish, M. and Kruschke, J. Decision boundaries in one dimensional categorization. Journal of Experimental Psychology: Learning, Memory and Cognition 23, 6, 1362-1377. 1997.
[38] Kruschke, J. Base rates in category learning. Journal of Experimental Psychology: Learning, Memory and Cognition 22(1): 3-26. 1996.
[39] Benerecetti, M.; Bouquet, P.; Ghidini, C.; Contextual Reasoning Distilled. *Journal of Theoretical and Experimental Artificial Intelligence*, 12(3):279-305, 2000.
[40] Aamodt, A.; Plaza, E. Case based reasoning: foundational issues, methodological variations and system approaches., *AI Communications*. IOS Press, Vol.7:1, pp.39-59. 1994.
[41] Yamada, Y.; Inuzuka, N.; Seki, H.; MAP classification with a similarity measure, N. Ishii (Ed.), Proceedings of The IASTED International Conference on Artificial and Computational Intelligence (ACI 2002), pp. 155-160, 2002..
[42] Melnik, S.; García-Molina, H.; Rahm, E.; Similarity Flooding: A Versatile Graph matching algorithm and its application to schema matching. In Proceedings of 18th International Conference on Data Engineering (ICDE), San Jose, CA, 2002.
[43] Hasegawa, F.; Dos Santos, E. L.; Ávila, B. C. and Kaestner C. A. A.; An Overview of Memory: Some Issues on Structures and Organization in the Legal Domain. IV Workshop on Agents and Intelligent Systems. CACIC 2003.
[44] Plato: Cratylus (Dialog of Hemogenes and Socrates, about sayings by Cratylus)

[45] Barsalou, L.W.; Being There Conceptually: Simulating categories in preparation for Situated Action. In N.L.Stein, P.J. Bauer, & M. Rabinowitz (Eds.) *Representation, memory and development: Essay in honor of Jean Mandler*, Mahwah, NJ:Erlbaum. October 2000.
[46] Werner, Heinz; Kaplan, Edith; The Adquisition of word meanings: A developmental study. *Monograph of the Society for Research in Child Development*, Nº 51, 1952.
[47] Sternberg, R.J.; Powell, J.; Comprehending verbal Comprehension, American Psychologist, 38, 878-893. 1983
[48] Gauker, Christopher, *Words without meaning*. MIT Press, Cambridge Massachusetts, 2003.
[49] Sternberg; R. J.; Cognitive Pychology, Chapter 6 Knowledge Representation and Information Processing, p.196-220, Harcourt Brace College Publishers, 1996.
[50] Jarmasz, M.; Szpakowicz, S. Roget's Thesaurus and Semantic Similarity. Proceedings of Conference on Recent Advances in Natural Language Processing (RANLP 2003), Borovets, Bulgaria, 212-219, September 2003.
[51] Smith, E.E.; Medin, D.L.; Categories and Concepts. Cambridge, MA, Harvard University Press, 1981.
[52] Guarino, N., and Welty, C.;Towards a methodology for ontology-based model engineering. In Proceedings of the ECOOP-2000 Workshop on Model Engineering. Available. 2000.
[53] Artale, Alessandro; Franconi, Enrico; Guarino, Incola; Open Problems for Part-Whole Relations. 1996 international Workshop on Description Logics (DL'96), Boston MA, November 1996.
[54] Edmonds, P; Hirst, G. Near Synonyms and Lexical Choice. *Computational Linguistics*, Volume 1, Number 1, Asociations of Computational Linguistics. 2002.
[55] Chi, M.T.H.; Roscoe, R.D. The Processes and Challenges of Conceptual Change. In Limon and Mason (Eds.) *Reconsidering Conceptual Change: Issues in Theory and Practice*, Kluwer Academic publishers, pp 3-27, 2002.
[56] Resnik, P. Semantic classes and syntactic ambiguity. In *Proceedings of the 1993 ARPA Human Language Technology Workshop*. Morgan Kaufmann, March 1993.
[57] Resnik, P. Semantic Similarity in a Taxonomy: An information based measure and its application to Problem of ambiguity in Natural Language. *Journal of Artificial Intelligence*, 1998.
[58] Sowa, J. F., Knowledge Representation: Logical, Philosophical, and Computational Foundations, Section 2.3 Top-Level Categories. Brooks Cole Publishing Co., Pacific Grove, CA, 2000.
[59] Tversky, A., and Kahneman, D., Judgement under uncertainty: Heuristics and biases, In *Judgement under uncertainty: Heuristics and biases*, Kahneman, Slovic and Tversky Editors. Cambridge University Press, 1982. Reprint 2001.
[60] Pierre de Fermat. Lettre de Blaise Pascal à Pierre de Fermat. En *Oeuvres*, 29 juillet 1654.
[61] Phillips, S., Halford, G. S., & Wilson, W. H. The processing of associations versus the processing of relations and symbols: A systematic comparison. Proceedings of the Seventeenth Annual Conference of the Cognitive Science Society, Pittsburgh,PA, 688-691, 1995.
[62] Woods, W.; What's in a Link: Foundations for Semantic Networks, in D.G. Bobrow & A. Collins (eds.), *Representation and Understanding*, Academic Press; reprinted in, Collins & Smith (eds.), Readings in Cognitive Science, section 2.2. 1975.
[63] Kahneman, D.; Tversky, A.; Choices, Values and Frames, *In Choices Values and Frames*, Daniel Kahneman and Amos Tversky Editors, Cambridge University Press, Reprinted, 2002.
[64] Corter, J.E. Similarity, confusability, and the density hypothesis. J. Exp. Psychol.: General. 116:238-49. 1987.
[65] Quillian, M.R. Semantic Memory, In Minsky, M. Ed., *Semantic Information Processing*. MIT Press, Cambridge. MA. 1968
[66] Collins, A.; Loftus, E. A spreading activation theory of semantic processing. *Psycologycal Review*, 82, 407-428. 1975.
[67] Sowa, J.F.; Conceptual Structures: Information Processing in Mind and Machine, Addison-Wesley, Reading, MA 1984
[68] Horwich, Paul; *Truth*, Basil Blackwell Ltd, Oxford, 1990.
[69] Theodorakis, M.; Analyti, A.; Constantopoulos, P.; Spyratos, N.; Contextualization as an Abstraction Mechanism for Conceptual Modeling. 1999.
[70] Euzenat, Jérôme; Towards a principled approach to semantic interoperability, Proc. IJCAI workshop on Ontologies and information sharing, Seattle (WA USA), 2001.
[71] McCarthy, J.; Generality in Artificial Intelligence. Communications of ACM, 30(12):1030-1035, 1987.
[72] Stuckenschmidt, Heiner. Using OIL for Intelligent Information Integration. In V. Benjamins, A. Gomez-Perez, and N. Guarino, editors, Proceedings of the Workshop on Applications of Ontologies and Problem-solving Methods, 14th European Conference on Artificial Intelligence ECAI 2000, Berlin, Germany, Aug. 21 -- 22, 2000.

A Fuzzy Querying System based on SQLf2 and SQLf3

Juan Eduardo
Universidad Simón Bolívar, Departamento de Computación,
Apartado 89000, Caracas 1080-A, Venezuela
jceduardo@hotmail.com

and

Marlene Goncalves
Universidad Simón Bolívar, Departamento de Computación,
Apartado 89000, Caracas 1080-A, Venezuela
mgoncalves@ldc.usb.ve

and

Leonid Tineo
Universidad Simón Bolívar, Departamento de Computación,
Apartado 89000, Caracas 1080-A, Venezuela
leonid@ldc.usb.ve

Abstract

For improving the expressive power, there have been proposed and developed several extensions of SQL. One of them is SQLf, a fuzzy set based extension that allows the expression of flexible queries involving user preferences. On the other hand, SQL has evolved into: SQL2 that incorporates relational algebra operations constrains and sub-queries; and SQL3 that incorporates features of deductive, active and object oriented databases. In a previous work we have defined SQLf2 and SQLf3 as extensions of SQLf with the fuzzy set based treatment of new features from SQL2 and SQL3. In this paper we present a real fuzzy querying system based on SQLf2-SQLf3 that we have built on top of a RDBMS. This system provides a web based interface and an API.

Keywords: Queries, fuzzy queries, database, web interface, SQL.

1. INTRODUCTION

The use of boolean logic restricts the treatment of imprecision or uncertainty in database systems. The queries don't get prospective results, because these queries are restrictive and they don' t contemplate important elements that could be considered to analyze data. A solution to this problem is using fuzzy sets in databases systems in order to express non crisp data and gradual (or flexible) requirements. We have concentrated our efforts in the study and implementation of fuzzy querying languages on relational databases due to the fact that many organizations store their information in relational databases.

There are some different proposals of fuzzy set based flexible querying languages, SQLf [[3]] is one of them that has been proposed by the research team of IRISA-ENSSAT, whose under the leadership of Patrick Bosc. SQLf is a fuzzy extension of the standard SQL that allows using fuzzy conditions in any place where SQL allows a regular one. We have adopted this proposal since 1998 [[26]] due to its desired characteristics. We have proposed some extensions to SQLf [[25]] and we have work in a real implementation of a flexible querying system based in SQLf and its extensions. We call this query system SQLfi (SQLf on Internet).

The standard SQL has evolved due to the new technology present in database systems, ISO and ANSI organizations have carried out the work of updating the standards definition of SQL. The two later versions of SQL are known as SQL2 [[1]] and SQL3 [[20]]. The norm SQL2 incorporates some relational algebra operations, some mechanisms for constrains specifications and the possibility of subqueries use as the base relation for a query. The norm SQL3 incorporates some features of deductive databases, active databases and object oriented databases.

In previous works we have studied the new features of SQL2 and SQL3 determining which of theme are susceptible of a fuzzy treatment. These works leaded us to the definition of two extensions of SQLf, named: SQLf2 [[10]] and SQLf3 [[11]], corresponding to the definition of fuzzy querying components based in the norms SQL2 and SQL3, respectively. We think that it is not sufficient to contribute to the theme of fuzzy querying proposing the theoretical definition of such languages, but also making a real implementation of them in order to allow further studies of behavior and development of applications.

We have structured the present paper as follows: Section 2 briefly presents some other' s related works, in section 3 we present the fuzzy query languages SQLf, SQLf2 and SQLf3. On the other hand, section 4 tells how our system prototype has evolved throw the intervention of some collaborators. Section 5 is devoted to outline the new SQLf2 and SQLf3 features that are implemented in the current system prototype, we omit their features proper of SQLf that were implemented in previous versions of the prototype. The architecture of our fuzzy querying system is shown in section 6. Some remarks about the used implementation tools are in section 7. Finally, we summarize and point out to the future in the section 8.

2. RELATED WORKS

Several efforts have been made by different authors in order to provide flexible querying database systems; some of them are the following:

OMRON [[21]] is a processor that contains a variant of SQL with fuzzy logic and is a fuzzy information retrieval library based. It' s a fuzzy query interface on traditional databases[[19]]. Don' t allow fuzzy quantifiers use.

FQUERY is an effort that adds fuzzy query functionality on a small database management system [[14]] (Microsoft ACCESS for Windows). Allow fuzzy quantifiers use in order to qualify criteria quantity that satisfies a query, but this is not allowed in partitioned or nesting queries.

BANDINS is a project that allows data flexible representation and manipulation. Its query language is SQLf [[3]]. There are some query processing mechanism [[4]][[5]][[6]][[7]] and the most relevant is derivation principle [[6]][[7]] because it has a less cost.

Fukami-Umano in [[27]] proposed a fuzzy relational model. Which query language is the Fuzzy Relational Algebra. They propose the building of a new database engine with fuzzy querying capabilities.

A fuzzy query language was presented by Wong-Leung in [[28]] to retrieve information by means of translation from a fuzzy query to a query for the database management system VAX Rdb/VMS and multicriteria decision making.

ISKREOT (Intelligent System for Knowledge Representation using Expert system and Object Technology) [[18]] is presented in and it is front-end intelligent information interface operating through a relational database ORACLE with fuzzy queries.

None of these previous works have incorporated in the proposed and developed querying systems features of SQL2 and SQL3 with a fuzzy set based treatment

3. QUERY LANGUAGES

One of the more remarkable efforts in the creation of Flexible Querying System for Relational Databases is SQLf [[3]]. It is the most complete fuzzy extension [[26]] to SQL due to the diversity of fuzzy queries that allows the extension of all SQL constructions with Fuzzy Logic.

Fuzzy queries involve fuzzy terms (atomic predicates, modifiers, connectors, comparators and quantifiers) whose meaning is user and context depending. Therefore a SQLf based querying system must provide a language for defining such terms.

In SQLf, the querying basic block is:

SELECT <attributes> FROM <relations>
WHERE <fuzzy condition> WITH CALIBRATION [n|λ|n,λ]

Semantic of this construction is cartesian product of relations involved in clause FROM, selecting those tuples that satisfied fuzzy condition and taking the fuzzy projection of those attributes indicated in clause SELECT. In clause WHERE, some logical expression can be used formed with terms defined by the user and predefined operators of fuzzy logic.

The result of a query in SQLf is a fuzzy set. In clause WITH CALIBRATION, a tolerance is specified to select tuples, which is called calibration. In the original definition of Bosc and Pivert [[3]], the calibration is specified in the clause SELECT, in a later work, Goncalves and Tineo [[10]][[11]] extended the definition with this clause in proof the orthogonality of language. Two calibration types exist: Quantitative and Qualitative. In case of quantitative calibration, a maximum number "n" of answers is obtained and in case of qualitative calibration those tuplas whose membership value is bigger or similar to the minimum level of tolerance "λ" is obtained.

SQLf also allows subqueries with different nesting operators. Subqueries are query structures like the basic block without the WITH CALIBRATION clause. Into the subqueries, attributes from the relations in the outer query may be used both in the SELECT and the WHERE clause. The general structure of such queries is:

SELECT <attributes> FROM <relations>
WHERE <nesting operator> <fuzzy subquery> WITH CALIBRATION [n|λ|n,λ]

SQLf also has a partitioned block querying structure. In this kind of query it is possible to impose a selection fuzzy criterion over the groups of the partition. Such queries are:

SELECT <attributes> FROM <relations> GROUP BY <attributes>
HAVING <fuzzy condition> WITH CALIBRATION [n|λ|n,λ]

Inspired in the norm SQL2, we have defined SQLf2 [[10]], an extension of SQLf that contemplates incorporation of fuzzy set based extensions of: relational algebra operators, integrity constraints, views definitions, support of date and time types, subqueries in from clause and data manipulation operations and new conditions kinds.

On the other hand, we have created SQLf3 [[10]] as an extension of SQLf inspired in the norm SQL3. SQLf3 includes elements of deductive databases, active databases and object oriented databases. They all with a fuzzy set based treatment.

For simplicity and space restrictions, we don' t present here the features of SQLf2 and SQLf3 in detail. The reader is referred to the previous works [[10]][[11]]. We will just enumerate the SQLf2 and SQLf3 features that where implemented in the built interpreter, see section 5

4. PROTOTYPE EVOLUTION

Since 1998, Tineo [[25]][[26]] has been directing the creation of a flexible querying system to relational databases based in SQLf. This system has been developed on top of the ORACLE Relational Database Management System (RDBMS). The development has been made in the database laboratory at Simón Bolívar University and Venezuela' s Central University.

As all software, SQLf has experienced some iteration in its life cycle in order to increasing its functionality. Some times, the prototype was limited to existing technology and enhanced in a new version.

In 1999, Borrajo and Rengifo [[2]] carried out a first prototype of an interpreter with a minimum subset of SQLf. This prototype was limited to the type of queries expressed with a simple block, and it was developed with certain built in linguistic terms stored in a database.

In 2000, Gutiérrez [[12]] has performed the implementation of evaluation mechanisms for SQLf. He has dealt with the three kinds of queries of SQLf: Basic Block, Nesting and Partitioning. This implementation was restricted to specific queries programmed in the code, based in a real database.

Also in 2000 Ramírez [[24]] extended the prototype integrating the mechanisms, allowing interactive definition of fuzzy terms. So this new prototype satisfied the complete SQLf. With it, the user may use his proper database and define his proper linguistic terms.

In 2001 Goncalves [[9]], extended the existing prototype of SQLf with some new characteristics defined for SQLf2 and SQLf3. Nevertheless, Goncalves' work has been focused mainly in the theoretic definition of new extensions rather than the completion of the prototype.

In 2002 Hernández, Montaña [[13]] and León, Martínez [[15]] have worked in the extension of the existing prototype with the new versions of the querying language: SQLf2 and SQLf3. They have implemented some of characteristics that had not been carried out in the previous version.

Also in 2002 Rodríguez and León [[16]] developed an improved new prototype of flexible querying system with all SQLf characteristics. This new prototype is based on Internet technology, so it's called SQLfi.

In this paper we present the ultimate SQLfi prototype, which includes the characteristics defined for SQLf2 and SQLf3. This implementation provides an application program interface in order to be used in some languages as ASP, JSP, JAVA, among others [[8]].

5. NEW FEATURES

Current system prototype implements most the new features of SQLf2 and SQLf3 functionality. The used developing tools limited the selection of the implemented features. We have chosen those features that were compatible with the support offered by the used RDBMS, Oracle 8i.

5.1 SQLf2 Features

The implemented SQLf2 characteristics are:

1. Fuzzy comparison over date type: Support of fuzzy comparison operators for Date type.
   CREATE COMPARATOR <symbol>
   ON {DATE | TIMESTAMP} AS <expression>
2. Checks with λ calibrated fuzzy conditions: It's possible to having sub-queries with calibration in the clause CHECK of CREATE TABLE sentence.
   CHECK (<fuzzy- condition>)
   WITH CALIBRATION λ
3. Views based in fuzzy queries: Support of creating and modifications on views defined by selection with calibration of some attributes of a relationship.
   CREATE VIEW <name> AS <subquery>
   WITH CALIBRATION λ
4. Constraints that involves multiple tables: The constraints are defined by means of CHECK clause. It is sometimes required that a constraint involves a complete relationship or more than a relationship.
   CREATE ASSERTION <name> CHECK < fuzzy- condition >
   WITH CALIBRATION λ
5. Unique operator over fuzzy conditions in partitioning: Unique operator indicates if duplicates exist in a fuzzy sub-query.
   SELECT <attributes> FROM <relations> GROUP BY <attributes>
   HAVING UNIQUE <fuzzy condition>
   WITH CALIBRATION λ
6. Fuzzy selection control structure: Support of case constructor with fuzzy comparisons.
   SELECT CASE (<attributes>)
   WITH CALIBRATION λ
   {WHEN <fuzzy predicate> THEN action}
   FROM <relations> WHERE <fuzzy condition>
7. Join over the result of fuzzy queries: Support of cross and natural join with calibration and fuzzy condition.
   (<subquery> {CROSS | NATURAL} JOIN <subquery>)
   WITH CALIBRATION λ
8. Theta joins operators over the result of fuzzy queries: Support of outer join with calibration and fuzzy condition
   (<subquery> [[LEFT | RIGHT | FULL] OUTER] JOIN <subquery>
   ON <fuzzy condition>)
   WITH CALIBRATION λ
9. Queries over the table resulting from a fuzzy sub-query: Support of a fuzzy sub-query o join in FROM clause.
   SELECT <attributes> FROM <subquery>
   WHERE <fuzzy condition>
   WITH CALIBRATION λ
10. Set operators over the result of fuzzy queries: Support of set operators for fuzzy query.
    (<sub-query> {UNION | INTERSECT | EXCEPT} <subquery>)
    WITH CALIBRATION λ
11. UPDATE operation with fuzzy condition: Support of UPDATE operators with fuzzy sub-queries on the same table.
    UPDATE <table> SET <attrib>=<value> [{, <attrib>=< value>}]
    WHERE <fuzzy condition>
    WITH CALIBRATION λ
12. DELETE operation with fuzzy condition: Support of DELETE operators with fuzzy sub-queries on the same table
    DELETE FROM <table> WHERE <fuzzy condition>
    WITH CALIBRATION λ

5.2 SQLf3 Features

The implemented SQLf3 characteristics are:

1. Fuzzy predicates on complex structures: Definition of fuzzy predicates on complex structures and datatypes defined by the user.
   CREATE FUZZY PREDICATE <name> ON <tuple type> AS <fuzzy condition>
2. Triggers with fuzzy conditions: Support of fuzzy trigger with components of traditional trigger (an action and a condition) viewed as fuzzy extension.
   CREATE TRIGGER <name> … WHEN (<fuzzy condition>) WITH CALIBRATION λ {<action>}
3. Fuzzy conditions in control structure IF: Fuzzy extension of IF sentence of the language procedural.
   IF <fuzzy condition> THEN {<action>}
   [{ELSEIF <fuzzy condition> THEN {< action >}}]
   [ELSE {< action >}]
   WITH CALIBRATION λ
   END IF
4. Fuzzy conditions in control structure FOR: Fuzzy extension of FOR sentence of the language procedural.
   FOR <result> AS <fuzzy query> DO <action> END FOR
5. Fuzzy conditions in control structure WHILE: Fuzzy extension of WHILE sentence of the language procedural.
   WHILE <fuzzy condition> DO {<action>} WITH CALIBRATION λ END WHILE
6. Fuzzy conditions in control structure REPEAT: Fuzzy extension of REPEAT sentence of the language procedural.
   REPEAT {<action>} UNTIL <fuzzy condition> WITH CALIBRATION λ END REPEAT
7. Fuzzy quantifiers FOR SOME and FOR ALL: To extend FOR ALL and FOR SOME so that they operate on fuzzy sub-queries.
   SELECT <attributes> FROM <relations>
   WHERE [FOR SOME | FOR ALL] <attribute> <fuzzy subquery>
   WITH CALIBRATION λ

6. SYSTEM ARCHITECTURE

For building the fuzzy querying system, we have used a three layers' architecture (Fig. 1). The lowest, data layer, is the RDBMS. The middle, logical layer, is an interpreter of SQLf2-SQLf3. The upper layer is the client's interface.

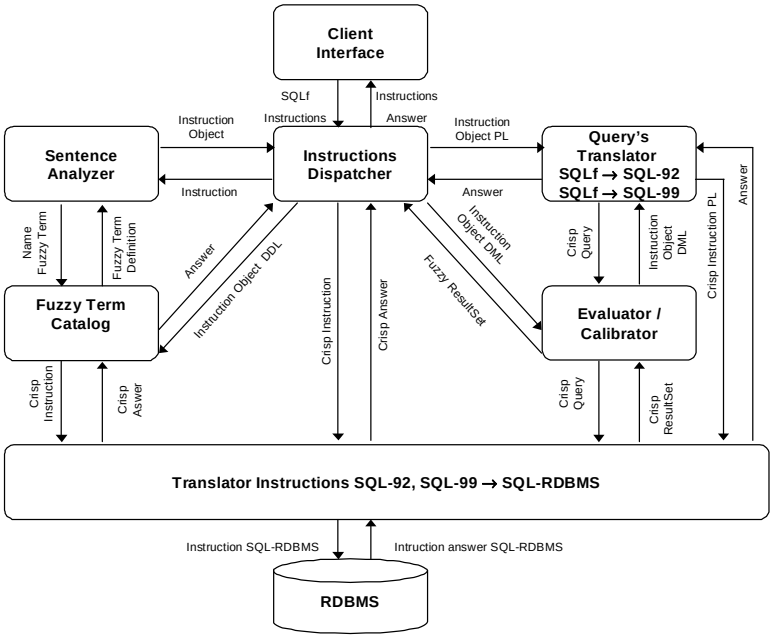


Fig. 1. Fuzzy Querying System Architecture

The main components inside the querying system are:

Client's Interface. Receives user' s fuzzy queries and term definitions. Shows the final results of user operations: the fuzzy query answer.

Instructions Dispatcher: It is the responsible for delivering at remainder of the modules the necessary structures for the execution of the instruction and to receive of these the respective answers.
Fuzzy Terms Catalog. Allows the specification of user defined fuzzy terms. Retrieves the definition of such terms in order to be used by the sentence analysis and evaluation mechanisms. These terms are stored into a database.
Sentence Analyzer. Analyzes the sentences introduced by the user, checking the syntactic and semantic correctness of the statements. Perform the translations needed for the execution of the statements. In case of fuzzy queries, builds a tree structure of the fuzzy query that is used in the evaluation process.
Evaluator/Calibrator. Performs the evaluation of fuzzy queries. Interacts with the RDBMS in order to retrieve database element relevant to the query processing. For so doing, uses a regular SQL query that is given by the Sentence Analyzer. Computes the satisfaction degrees and calibrates the answer of the fuzzy query. For so doing uses the result of the regular query and the tree structure of the fuzzy query.
Query's Translator SQLf → SQLf92 and SQL99. It is the responsible for translating queries that contain fuzzy terms in regular queries using the derivation principles, producing queries in SQL-92 and SQL-99 standards.
Translation Instructions SQL-92 and SQL-99 → SQL-RDBMS. We have adopted to express regular queries in the standards SQL-92 and SQL-99 in order to make our system as portable as possible. Therefore, we need to translate regular queries into specific RDBMS query language.

7. IMPLEMENTATION

This querying system has been made using:
The Java programming language [[22]]. Java is a language of programming object oriented for a distributed application on WEB platform. Besides it offers the facility of libraries and/or interfaces for connection with databases or server pages using JDBC convention.
The ORACLE 8.0.5.0.0 [[15]] [[17]]. RDBMS. Oracle is one of the most efficient in the market for storage capacity, answer capacity, stability, backup mechanisms, security, etc. The tool of Oracle used was OCI (Oracle Call Interface) that is an interface for programming of applications that allows to the applications written in C interact with an Oracle server.
The CLAIRE [[23]] is a 100% Java compatible LR (1) parser and lexical analyzer generator, designed and developed by programming language research group at Simón Bolívar University. It's also language independent that is it can generate code on any language it has a plug in for. The resulting parser is a Java program module.

8. CONCLUDING REMARKS

In this paper is shown the feasibility of implementation of a fuzzy querying system on web that embraces most of the characteristics of SQLf2 and SQLf3, enriching the functionality of the existent prototype of SQLf with a bigger quantity of requirements that the user can execute. Also, it was appealed to a series of tests whose results guaranteed that this prototype adapts to the requirements and necessities outlined for the support of the new characteristics of SQLf2 and SQLf3.
The built fuzzy querying system supports: fuzzy comparison on dates. fuzzy checks. Partitioned queries using the UNIQUE operator over fuzzy conditions, fuzzy views, fuzzy assertions, fuzzy joins, fuzzy subqueries in the from clause, fuzzy set operations, update and delete with fuzzy conditions, fuzzy predicates on complex structures, control instructions with fuzzy conditions, triggers with fuzzy conditions in the when clause, stored functions, procedures and ADT's with fuzzy elements.
We are working now in the implementation of the remaining SQLf2 and SQLf3 features. We also points to performance studies of system prototype and definition of different query evaluation mechanisms for fuzzy queries in SQLf2 and SQLf3.
We know a fuzzy query can return a bigger quantity of results that a classic query. The most remarkable fuzzy queries process mechanisms are based in *λ-cut*, by the application of a principle called Derivation Principle, for tkin advantages of the relationship between fuzzy queries and regular ones. This principle has been defined and studied by Bosc and Pivert [7] and Tineo [7]. We work in this principle to SQLf2 and SQLf3 and its implementation as query evaluation mechanism in our prototype.

References

[1] ANSI X3. Database Language SQL, 135-1992, American National Standards Institute, New York.

[2] Borrajo, F., Rengifo, G., Implementación del Lenguaje de Interrogaciones Flexibles a Base de Datos Relacionales SQLf, Informe final de Proyecto de Grado, Universidad Simón Bolívar., Julio 1999.

[3] Bosc P. and Pivert O. SQLf: A Relational Database Language for Fuzzy Querying, IEEE Transactions on Fuzzy Systems, Vol 3, No. 1, Feb 1995.

[4] Bosc P. and Brisson A. On the evaluation of some SQLf nested queries, Proceeding International Workshop on Fuzzy Databases and Information Retrieval, 1995.

[5] Bosc P., Pivert O. and Farquhar K. Integrating Fuzzy Queries into an Existing Database Management System: An Example, International Journal of Intelligent Systems, V.9, pp 475-492,1994

[6] Bosc P. and Pivert O. On the efficiency of the alpha-cut distribution method to evaluate simple fuzzy relational queries, Advances in Fuzzy Systems-Applications and Theory, Vol 4, Fuzzy Logic and Soft Computing, B. Bouchon-Meunier, R.R.Yager, L.A. Zadeh eds, Wold Scientific, pp 251-260, 1995.

[7] Bosc, P. & Pivert, O., SQLf Query Functionality on Top of a Regular Relational Database Management System, Knowledge Management in Fuzzy Databases, Pons, O., Vila, M. and J. Kacprzyk (Eds.), Physica-Verlag, (2000), Pp. 171-190.

[8] Eduardo J., Sistema de Interrogación Flexible a Bases de Datos SQLf2 – SQLf3. Informe Final de Trabajo Especial de Grado presentado ante la Universidad Simón Bolívar, Diciembre 2003.

[9] Goncalves M., Extensión del Lenguaje de Interrogación Flexible a Bases de Datos SQLf mediante las normas SQL2 y SQL3. Informe Final de Trabajo Especial de Grado presentado ante la Universidad Simón Bolívar, Septiembre 2001.

[10] Goncalves M. and Tineo, L. SQLf Flexible Querying Language Extension by means of the norm SQL2, The 10th IEEE International Conference on Fuzzy Systems, Vol 1, Dec 2001.

[11] Goncalves, M. and Tineo, L. SQLf3: An extension of SQLf with SQL3 features, The 10th IEEE International Conference on Fuzzy Systems, Vol 3, Dec 2001.

[12] Gutiérrez, L. Desempeño de Mecanismos de Evaluación de SQLf, Informe final de Proyecto de Grado, Universidad Simón Bolívar, Septiembre 2000.

[13] Hernández, G., Montaña A. HECDOCf: Una Herramienta para la Evaluación de Cursos y Docentes mediante SQLf2 a través del Web, Informe final de Proyecto de Grado presentado ante U.C.V., Junio 2002.

[14] Kacpryzyk J. and Zadrozny S., Fuzzy Queries in Microsoft AccessTM v.2, Proceedings of Fuzzy IEEE'95 Workshop on Fuzzy Database Systems and Information Retrieval, pp 61-66, 1995.

[15] León, G., Martínez, D. SISECDf3: Sistema de apoyo basado en la tecnología Internet para la Evaluación de Cursos y Docentes mediante SQLf3, Informe final de Proyecto de Grado presentado ante U.C.V., Junio 2002.

[16] León W., Rodríguez H., "Sistema de Interrogación Flexibles en Internet a Bases de Datos Relacionales SQLfi" Informe final de proyecto de grado presentado ante la U.S.B., Octubre 2002.

[17] Loney K. and Koch G. Oracle8i The Complete Reference. ISBN: 0072123648. McGraw-Hill. 2000.

[18] Loo G. and Lee K. An Interface to Databases for Flexible Query Answering: A Fuzzy-Set Approach. Lecture Notes in Computer Science 1873. DEXA 2000. Septiembre 2000. Londres, Reino Unido. pp 654-663.

[19] Mansfield W. & Fleischman R. A High-performance, Ad-hoc, Fuzzy Query Processing System Journal of Intelligent Systems, Vol. 2, pp. 397-420, 1993.

[20] Melton J. ISO/ANSI Working Draft: Database Language SQL (SQL3), X3H2-93-091/ISO DBL YOK-003.

[21] Nakajima H., Sogoh T., Arao M. Fuzzy Database Language and Library-Fuzzy Extension to SQL, Proceedings of Second IEEE International Conference on Fuzzy Systems, pp 477-482, 1983.

[22] Naughton P., Schildt H. Java Manual de Referencia, McGraw-Hill.

[23] Pacheco P., "Implementación del Analizador Lexicográfico, Sintáctico y Semántico Claire", Informe final de proyecto de grado presentado ante la U.S.B., Enero 2000.

[24] Ramírez, J. Interpretador del Lenguaje de Interrogaciones Flexibles a Bases de Datos Relacionales SQLf, Informe final de Proyecto de Grado, Universidad Simón Bolívar, Enero 2001.

[25] Tineo L. Algunos Aportes en Bases de Datos Difusas. Trabajo de Ascenso presentado ante la Universidad Simón Bolívar, Sartenejas Septiembre 2001.

[26] Tineo L. Interrogaciones Flexibles en Bases de Datos Relacionales. Trabajo de Ascenso presentado ante la Universidad Simón Bolívar, Sartenejas Enero 1998.

[27] Umano M., Fukami S. Fuzzy Relational Algebra for Possibility-Distribution-Fuzzy-Relational Model of Fuzzy Data, Journal of Intelligent Information System, Vol 3, pp 7-27, 1994.

[28] Wong M. and Leung K. A fuzzy Database-Query Language. Information Systems. Vol 15. No. 5, pp 583-590, 1990.

Simulación y Visualización de la Performance de un Administrador BSP

Paula A. Millado, Daniel O. Laguia, Albert O. Sofia
Universidad Nacional de la Patagonia Austral
Río Gallegos, Argentina (9400)
{pmillado;dlaguia;osofia}@unpa.edu.ar

and

Mauricio Marín
Universidad de Magallanes
Punta Arenas, Chile
mauricio.marin@umag.cl

and

Claudio Delrieux
Universidad Nacional del Sur
Bahía Blanca, Argentina
claudio@acm.org

Abstract

This paper describes a graphic tool for visualizing the behavior of a distributed database server. The target is to provide a graphic tool for the database administrator to evaluate the performance of the database and to suggest a tunning in the broker algorithm, in order to get better request times for the user's queries. We consider a server working with a large query traffic. This workload is served using parallel processing over a cluster with an implementation of the parallel computing BSP model. The visual tool we propose here, allows to show the amount of communication and synchronization time between processors needed by the parallel processing of the queries in the cluster and the workload in database. This information is represented with two visual metaphors, which are useful for the database administrator to take decisions. In this context, the query's order execution can have effects very different at the request time for each query.

Keywords: Databases, Parallel Processing on SQL Queries, Parallel and Distributen Computing, BSP, Cientific Visualization

Resumen

En este trabajo se describe una herramienta gráfica de visualización de la operación de un servidor de bases de datos distribuidas. El objetivo es proporcionar al administrador de la base de datos una herramienta que le permita observar el comportamiento y tomar decisiones orientadas a mejorar los tiempos de respuesta a consultas SQL generadas por los usuarios del sistema. Suponemos un servidor operando con una gran intensidad de tráfico de consultas. Dicha carga de trabajo es servida empleando procesamiento paralelo sobre un *cluster* de PCs, por medio de una implementación del modelo BSP de computación paralela. La herramienta permite visualizar aspectos tales como la cantidad de comunicación y sincronización entre procesadores demandada por el procesamiento paralelo de las consultas. Esa información, adecuadamente representada mediante metáforas visuales, es presentada al administrador para la toma de decisiones. Esto, debido al orden en que los diferentes tipos de consultas a la base de datos generadas por los usuarios son ejecutadas, puede tener efectos muy distintos en el tiempo de respuesta de cada consulta.

Palabras claves: Bases de Datos, Procesamiento Paralelo de Consultas SQL, Computación Paralela y Distribuída, BSP, Visualización Científica

[1]Este trabajo fue parcialmente financiado por la red CYTED-Ritos2 y por la SECyT-UNS

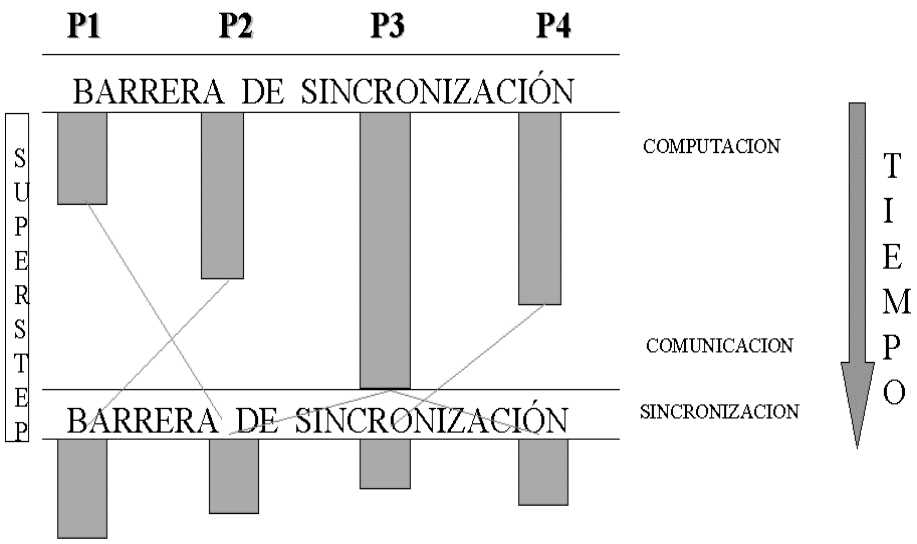


Figura 1: Modelo BSP y supersteps.

1 Modelo de computación paralela BSP

Varios productos comerciales fueron desarrollados para máquinas multiprocesadores de memoria compartida y distribuida [2, 4, 11], y más recientemente para *clusters* de computadores [6, 19, 20]. Todos estos desarrollos están basados en modelos tradicionales de computación paralela como paso de mensajes y memoria compartida. En este trabajo se presenta una solución basada en un modelo relativamente nuevo de computación paralela llamado BSP [21, 25, 28], el cual utiliza una configuración de base de datos distribuida para acelerar las consultas, realizando el procesamiento de la cola de consultas en forma secuencial.

En BSP [25] un computador paralelo es visto como un conjunto de procesadores con memoria local e interconectados a través de una red de comunicaciones de topología transparente al usuario. En este modelo, la computación es organizada como una secuencia de *supersteps*. Durante un superstep, cada procesador puede realizar operaciones sobre datos locales únicamente y depositar mensajes a ser enviados a otros procesadores. Al final del superstep, todos los mensajes son enviados a sus destinos y los procesadores son sincronizados en forma de barrera para iniciar el siguiente superstep. Es decir, los mensajes están disponibles en sus destinos al instante en que se inicia el siguiente superstep. Tal como lo indica la Fig. 1, un superstep está formado por una fase en que los procesadores realizan computaciones sobre datos almacenados en sus memorias locales, tiempo en el cual pueden depositar mensajes a ser enviados a otros procesadores, y al final del superstep se envían todos los mensajes y los procesadores se sincronizan con el objetivo de asegurar que todos han recibido los mensajes, y así comenzar el siguiente superstep.

2 Modelo de Predicción de Costos de BSP

La estructura del modelo BSP facilita la predicción del desempeño de programas y algoritmos. El costo de un programa está dado por la suma del costo de todos sus supersteps, donde el costo de cada superstep está dado por:

1. la suma del costo originado por las computaciones sobre datos locales;
2. el costo de las comunicaciones de mensajes entre procesadores;
3. el costo asociado a la sincronización de los procesadores.

Según lo explicado mas arriba, podemos componer el tiempo de cada superstep como la suma de tres términos: computación + comunicación + sincronización. En otras palabras, el tiempo de ejecución de un superstep s es la suma del máximo tiempo de procesamiento local, el tiempo de envío de los mensajes y el tiempo de sincronización [21, 16]:

$$\text{tiempo}(s) = \text{máx}\{w_i(s)\} + \text{máx}\{h_i(s)\} * g + l, \quad (1)$$

donde $w_i(s)$ es el tiempo de procesamiento local en un procesador i durante el superstep s y $h_i(s) = \text{máx}\{h_{i+}(s), h_{i-}(s)\}$, donde $h_{i+}(s)$ representa el numero de palabras transmitidas y $h_{i-}(s)$ representa el numero de palabras recibidas por el procesador i durante el superstep s. El tiempo de ejecución de un programa BSP está dado por la sumatoria de los tiempos de ejecución de cada superstep ($\sum_s \text{tiempo}(s)$). Por lo tanto el tiempo total de procesamiento puede representarse a través de la fórmula:

$$W + H * g + S * l, \tag{2}$$

donde

$$W = \sum_s \text{máx}\{w_i(s)\}$$

$$H = \sum_s \text{máx}\{h_i(s)\}$$

En general W, H y S son funciones de p (cantidad de procesadores) y n (tamaño del dato). Por lo tanto, el diseño de algotirmos BSP eficientes debe estar guiado por los siguientes criterios:

1. balance de computación en cada procesador, puesto que W es un máximo en computación;
2. balance en comunicación, puesto que $h_{i+}(s)$ y $h_{i-}(s)$ es un máximo sobre los datos enviados o recibidos; y
3. minimización del número total de supersteps requerido para completar el programa.

Los valores de l y g se obtienen en cada implementación de forma empírica, donde l representa el costo de la sincronización y g el costo de transmitir un word desde un procesador a otro en una situación de tráfico continuo en la red de comunicaciones.

3 Configuración del Servidor de Consultas

Actualmente es usual encontrar sitios Web que para su funcionamiento proporcionan acceso a bases de datos relacionales y de texto en la modalidad de cliente-servidor. En tales casos, es deseable que en el lado servidor, el administrador de la base de datos sea capaz de procesar lo más rápido posible las consultas producidas por un número grande de clientes generando peticiones de *Common Gateway Interface* (CGI) a través de lenguajes de *script* tales como PHP o Perl. En particular, un esquema típico puede ser una configuración donde exista una máquina *front-end* que reciba requerimientos desde clientes que ejecutan scripts, la cual a su vez envía instrucciones al servidor de bases de datos y espera por una respuesta del mismo. Estas máquinas *front-end* son las denominadas generadores de consultas. El servidor de base de datos usualmente se comunica con los generadores de consultas por medio de un lenguaje de *querying* como el SQL. Este servidor, además, puede estar alojado en otra máquina y en realidad puede atender simultáneamente los requerimientos provenientes desde dos o más máquinas generadoras de consultas.

Para tal propósito, se ha propuesto anteriormente [10] un servidor de procesamiento de consultas paralelo basado en el modelo *BSP pub* [26], que utiliza una configuración de base de datos distribuida para acelerar las consultas, realizando el procesamiento de la cola de consultas en forma secuencial. Dentro de la gran cantidad de consultas generadas por los generadores de consultas existen diferentes tipologías, las cuales poseen características distintivas de acuerdo a los recursos solicitados, debido fundamentalmente al tiempo de procesamiento requerido y el acceso a la base de datos. Una consulta de alta complejidad podría retrasar sustancialmente a las consultas simples en el esquema de una cola de procesamiento secuencial. El propósito del presente trabajo es presentar una solución gráfica para la representación del desempeño de un servidor paralelo de procesamiento de consultas, que permita asimismo la administración de la cola de consultas a través de la asignación de prioridades y la manipulación del orden de las consultas de la cola. El diseño e implementación de servidores paralelos de sistemas de bases de datos relacionales es un tema que ha sido investigado ampliamente durante la última década [5, 3, 9, 12, 17, 18, 14, 1].

Otro aspecto importante es que nuestra solución se construye a partir de tecnología existente como lo es un servidor secuencial de bases de datos en cada máquina del cluster [27] y el uso de una biblioteca de comunicaciones BSP [26] para gestionar la comunicación y sincronización de las máquinas. Es decir, el costo de implementación y puesta en operación es muy bajo. Proponemos un esquema fácil de implementar y muy eficiente respecto de la alternativa secuencial. El uso de BSP en esta clase de aplicaciones aun no ha sido investigado en profundidad. Trabajos preliminares, principalmente teóricos, en este tema pueden ser encontrados en [13, 23, 22, 24]. En esta línea, este artículo proporciona un enfoque y evaluación práctica de forma visual para el modelo de computación BSP, para lo cual se ha desarrollado una herramienta que permite evaluar el desempeño y administrar la cola de consultas.

La optimización de la gestión de las consultas es crítica para un aprovechamiento global de los recursos, y también para garantizar que la arquitectura es adecuadamente modular y escalable. En el modelo BSP, esta optimización depende de la administración del tipo de consultas, generalmente determinada por la programación de un *broker* o agente de gestión. Dentro de la gran cantidad de consultas generadas por los generadores de consultas existen diferentes tipologías, las cuales poseen características distintivas de acuerdo a los recursos solicitados, debido fundamentalmente al tiempo de procesamiento requerido y el acceso a la base de datos. Una consulta de alta complejidad podría retrasar sustancialmente las consultas más simples en el esquema de una cola de procesamiento secuencial. Lamentablemente no hay manera de poder predecir y optimizar estáticamente la administración de las consultas, por lo que una asignatura pendiente es la determinación dinámica de los parámetros que sintonice adecuadamente el funcionamiento optimizado del sistema en una situación particular.

Por dicha razón, en este trabajo exploramos la implementación y uso de una herramienta de visualización para la representación del desempeño del servidor paralelo de procesamiento de consultas, que permita asimismo la administración de la cola de consultas a través de la asignación de prioridades y la manipulación del orden de las consultas dentro de la misma. Para ello, se ha desarrollado un servidor de bases de datos paralelo utilizando el modelo de computación BSP, donde la base de datos se ha distribuido uniformemente. A ese servidor arriban consultas que se discriminan en dos tipologías básicas: consultas simples (correspondientes a *select* simples) y consultas complejas (correspondientes al *join* de dos o más tablas). Una descripción más detallada del diseño del servidor puede ser encontrada en [10, 15].

4 Configuración del Servidor Paralelo de Base de datos

Como ya mencionáramos, nuestra metodología de computación paralela opera sobre un cluster de PCs. El cluster es más bien un accidente que una necesidad puesto que BSP también esta disponible para sistemas multiprocesadores de memoria compartida y distribuida, y los programas pueden ejecutarse sin cambios en cualquiera de estas tres plataformas. En BSP, un computador paralelo es visto como un conjunto de procesadores con memoria local e interconectados a través de una red de comunicación de topología transparente al usuario. La realización práctica de este modelo es a través de una biblioteca de comunicaciones llamada *BSPlib*. Las primitivas de comunicación se invocan desde programas en C y C++, con funciones tales como `bsp_send()` y `bsp_sync()` para comunicar datos y sincronizar las máquinas respectivamente. El modo de programación es SPMD *(simple program multiple data)*. Es decir, un programa BSP es iniciado por el usuario desde una máquina y éste se duplica automáticamente en las máquinas restantes que conforman el cluster. Cada copia ejecuta los mismos superstep pero cada uno opera sobre sus propios datos locales. Adicionalmente, las copias pueden ejecutar caminos alternativos dentro del código de un superstep, pero todos deben alcanzar el mismo punto de sincronización (fin del superstep indicado con `bsp_sync()`) antes de continuar con el siguiente superstep. En nuestro caso, los datos locales son los trozos de la base de datos que se encuentra uniformemente distribuida en los discos de las máquinas que conforman el cluster. Las consultas SQL y las tablas resultado son transmitidas de una máquina a otra mediante arreglos de caracteres enviados utilizando `bsp_send()` y recibidos con `bsp_get()` [21].

La implementación del servidor es como sigue. Existe un conjunto P de procesadores ejecutando los superstep de BSP. En cada máquina existe un administrador de bases de datos relacionales (MySQL en nuestro caso). Este administrador es operado mediante instrucciones en lenguaje SQL enviadas a través de una conexión *socket* implementada por una API para C++. Por ejemplo, cada máquina del cluster puede ejecutar comandos SQL sobre su base de datos local mediante instrucciones tales como la siguiente que definimos como *consulta simple*:

```
Connection C("database");
Query Q = C.query();
Q << "select * from PRODUCTOS where disciplina=mensaje.preg";
Result R = Q.store();
cout << "Número total de tuplas recuperadas = " << R.rows() << endl;
```

El siguiente tipo de consulta se define como *Consulta Compleja*:

```
Connection C("database");
Query Q = C.query();
Q << "select sum(cantidad) from PRODUCTOS, VENTAS where PRODUCTOS.codigo= VENTAS.codigo AND
disciplina=mensaje.preg";
Result R = Q.store();
```

```
cout << "Número total de tuplas recuperadas = " << R.rows() << endl;
```

Cada una de las P máquinas mantiene el mismo esquema de la base de datos, es decir, las mismas tablas pero con distintas tuplas. Las tuplas están distribuidas uniformemente en la base de datos. Esto se hace mediante una función como *código mod P*, donde *código* es la llave de la tabla. Así, si existen N tuplas para una tabla global dada, estas se encuentran uniformemente distribuidas en las sub-tablas almacenadas en las P máquinas, cada una de ellas manteniendo aproximadamente $\frac{N}{P}$ tuplas. Las tablas no contienen tuplas duplicadas que estén ubicadas en la misma máquina o en otras. En nuestro caso, el generador de consultas va enviando instrucciones SQL de manera circular a cada máquina del cluster BSP. Cuando una máquina i recibe una instrucción SQL, esta transmite a todas las otras una copia de ella para que todas ejecuten la instrucción en sus respectivas bases de datos locales. Luego todas las máquinas transmiten a la máquina i sus resultados parciales de manera que la máquina i pueda construir el resultado final (tabla resultado) y enviárselo como respuesta al generador de consultas. Una implementación BSP de esta estrategia está descripta en el siguiente pseudo-código:

```
// Algoritmo básico ejecutado por cada máquina del cluster.
// Procesamiento paralelo de instrucciones select-from-where.
Inicializa cola de consultas SQL.
while( true )
Comenzar

 Recibe nuevos mensajes desde el generador de consultas SQL, y los almacena en la cola de consult

 Retira un mensaje desde la cola y lo envía a cada máquina del cluster (incluida esta misma,
     cada mensaje contiene el tipo de consulta y una identificación de la máquina que lo envía).

 BSP_SYNC(); (Fin del superstep, sincroniza las máquinas).

 Recibe mensajes desde las máquinas conformando el cluster.

 Por cada mensaje recibido, ejecuta la consulta SQL en el trozo de base de datos almacenados
     en esta máquina.

 Envía la sub-tabla resultante de la consulta a la máquina que originó el mensaje SQL.

 BSP_SYNC(); (Fin del superstep, sincroniza las máquinas).

 Recibe las sub-tablas y las integra para formar la tabla resultado asociada a la consulta
     iniciada por esta máquina.

 Envía la tabla resultado al generador de consultas.

Fin
```

Una característica de las aplicaciones de bases de datos en la Web, es que las tablas resultado que se envían a los clientes en formato HTML son generalmente de tamaño reducido. Esto permite que dichas tablas puedan ser almacenadas en arreglos de caracteres (buffers) en memoria principal. Es decir, la comunicación entre máquinas del cluster, y entre máquinas y generador de consultas, puede ser soportada por buffers de memoria principal sin necesidad de recurrir a técnicas de manejo de almacenamiento secundario. No obstante, en caso de ser necesario transmitir tablas de gran tamaño se pueden destinar superstep adicionales para transmitirlas por partes; cada una de un tamaño igual al máximo espacio disponible en los buffers de comunicación de BSPlib. El algoritmo mostrado en el pseudo-código anterior permite la ejecución concurrente de operaciones de sólo lectura en la base de datos. No es necesario realizar ningún tipo de sincronización de accesos a los datos puesto que estos no sufren modificaciones.

Una observación importante respecto de nuestra metodología de procesamiento paralelo es que los superstep proporcionan un mecanismo global de sincronización de operaciones en el tiempo. Cada ciclo de ejecución de operaciones comienza con el retiro, en cada procesador, de una instrucción desde la cola de consultas SQL, lo cual es seguido por una comunicación global. El generador de consultas envía instrucciones SQL consecutivamente. El punto aquí es cubrir el mayor espectro posible sin llegar a tener que construir un administrador de bases de datos paralelo o modificar uno secuencial. Enfatizamos que la solución aquí propuesta utiliza tecnología existente de bajo costo, un DBMS secuencial como MySQL [27], un grupo de PCs conectados mediante un switch, C++ y una biblioteca de comunicaciones para el modelo BSP de computación paralela llamada BSPpub [26].

Por supuesto, la eficiencia de este esquema depende de una adecuada distribución de los datos en las máquinas que conforman el cluster. Para esto el modelo BSP proporciona una forma de cuantificar el costo en computación, comunicación y sincronización de programas BSP. En [16] proponemos una extensión a dicho modelo con el objeto de ayudar al diseñador de la base de datos en la decisión por una determinada distribución tanto de tablas como de tuplas de dichas tablas en los discos de las máquinas del cluster.

Anteriores experimentos con esta configuración han demostrado una mejora altamente significativa en la performance de las consultas. En [10] realizamos la evaluación del desempeño de esta configuración. Sin embargo, debería ser posible aún mejorar el rendimiento ajustando parámetros y características de la cola de consultas SQL, de manera de tomar en cuenta el tiempo de espera de los clientes que han generado las consultas para maximizar su satisfacción al realizar un acceso a la aplicación. En la siguiente sección presentamos una descripción de la herramienta desarrollada para evaluar gráficamente el desempeño del servidor de consultas y administrar ciertas características de la cola de consultas SQL. En todos los casos se utiliza un benchmark basado en dos tipos de instrucciones select, puesto que como se explicó anteriormente en esta sección, con este tipo de instrucciones es más difícil obtener buen desempeño en el caso paralelo puesto que demandan mayor comunicación (envío de tablas de resultados parciales) que instrucciones como insert y update, las cuales no generan tablas de resultados.

5 Optimización Dinámica

El propósito de la visualización del desempeño del sistema es que el administrador pueda diagnosticar visualmente un desbalance en la carga que está recibiendo cada procesador. En ese contexto, el objetivo de esta herramienta es, además de observar el comportamiento de la base de datos paralela en tiempo real, permitir la administración de la misma, de manera de poder mejorar la performance del sistema. Esta optimización dinámica se logra operando sobre la cola de consultas, trabajando básicamente con dos colas, correspondientes una a las consultas simples y otra a las complejas. En nuestro caso hemos permitido al operador la posibilidad de asignar prioridades en la ejecución de las consultas, mediante la definición de un esquema de proporcionalidad en la relación consultas simples/consultas complejas.

Esta proporción indica una prioridad debido a que, en caso de existir en la cola de consultas operaciones simples y complejas, los clientes que han solicitado una consulta simple se verían retrasados en forma muy considerable si existiera una sucesión de varias consultas complejas. Aunque estas últimas se hayan producido con anterioridad, sería un esquema desventajoso para los clientes con consultas simples si se realizara la cola solo en forma secuencial. En el otro extremo, posponer las consultas complejas hasta la finalización de todas las consultas simples podría implicar una demora excesiva para las primeras. Si bien ésta es una solución sencilla, podrían estudiarse otras alternativas para optimizar la performance de la base de datos. Sin embargo es dificultoso manipular en tiempo real los parámetros del modelo de costos BSP, ya que implica la modificación del código fuente de los programas utilizados en el servidor.

6 Metáforas Visuales

En trabajos anteriores [7], desarrollamos el primer prototipo de visualización del desempeño que permitía una correcta administración del servidor paralelo. En esta primera aproximación se representaba la performance global de los procesadores en los parámetros funcionales más importantes para el servidor (Tiempo de Procesamiento TP, Tiempo de Sincronización TS, Tiempo de Acceso a la Base de Datos TB y Tiempo de Comunicación TC). Sin embargo, esta percepción global de los sucesos producidos en la cola de consultas es inadecuada, debido principalmente a la falta de granularidad en la visualización de los supersteps. Por dicha razón, una de las mejoras necesarias consiste en poder observar y comparar esta información con la correspondiente a otros supersteps, obteniendo por lo tanto un historial de la performance del sistema desde una base temporal determinada. Para ello, en el presente trabajo desarrollamos dos nuevas maneras de visualizar la performance del sistema.

En la primera metáfora visual es una evolución natural de los trabajos anteriores (ver Fig. 2). Aquí se optó por representar el comportamiento del sistema graficando la performance de cada procesador en cada superstep. De esa manera, el comportamiento de cada procesador está representado con una torre, en la cual cada uno de sus pisos es un superstep. El color de dicho piso representa la performance del procesador dado. Esto se logró mediante la graficación de los valores de los parámetros funcionales en una paleta bidimensional que representa, en el modelo de costos del modelo BSP, la relación entre el tiempo de comunicación y sincronización respecto del tiempo de procesamiento (ver Fig. 3). Una carga balanceada se caracteriza por tener $TP + TB$ comparable a $TS + TC$, no importando si estos valores son bajos o altos. La paleta, por lo tanto, permite además visualizar esta correlación. Un procesador que en un superstep insume mayor $TP + TB$ se representa con un color naranja, mientras que si $TS + TC$ es mayor, el color tiende al

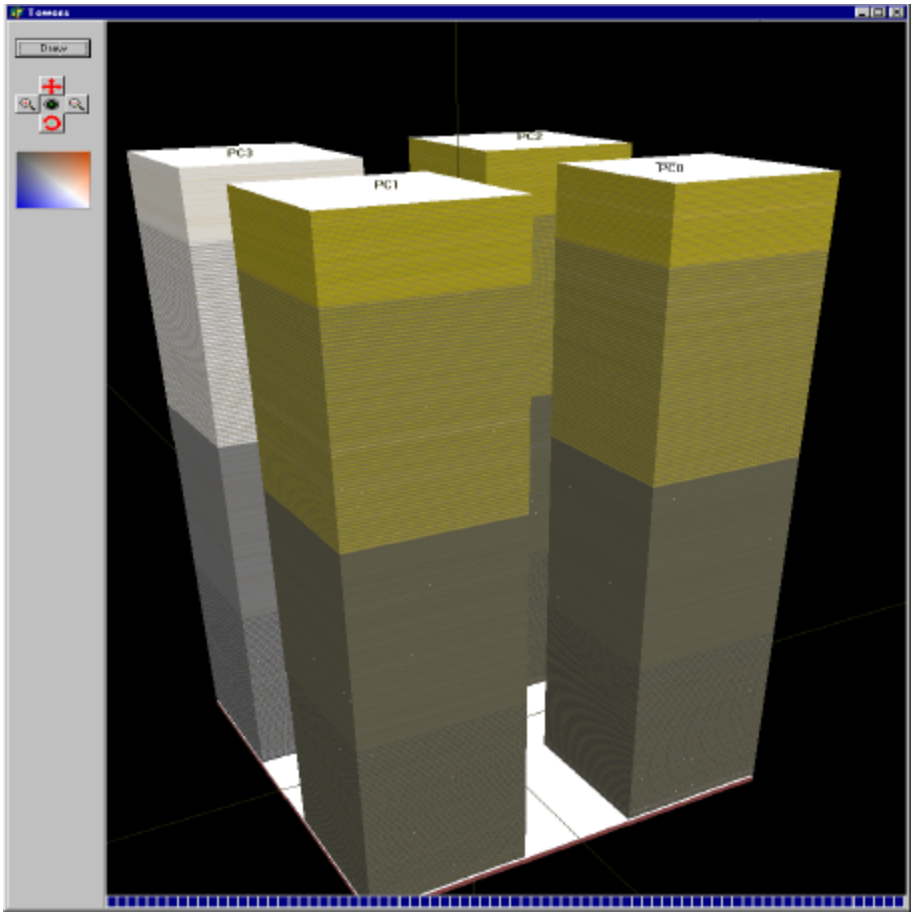


Figura 2: Sucesión de supersteps representada como *pisos* por orden de llegada. La performance de cada superstep se representa por medio de una paleta (ver Fig. 3).

azul. Si ambos tiempos están equilibrados, el color es neutro. Si ambos tiempos están equilibrados y son bajos, el color tiende al blanco, y si ambos son altos, tiende al negro. En la Fig. 2 se puede visualizar el cambio en la performance de los procesadores a medida que la proporción de consultas se va modificando.

Esta paleta bivariada se consigue asignando luminancia y saturación positivas en una de las variables a representar, y luminancia positiva y saturación "negativa" en la otra variable, interpretando la saturación negativa como saturación en la crominancia complementaria. De esa manera, valores altos de las variables sinergizan en producir una luminancia alta, pero compiten en crominancia, produciendo un color neutro. De hecho, la paleta muestra adecuadamente la zona de valores donde existe correlación entre ambas variables.

En base a la ecuación 2 puede obtenerse una relación entre el tiempo de ejecución de un superstep y el tiempo comparable de un procedimiento secuencial, dado por la siguiente fórmula:

$$\frac{T_{\text{secuencial}}}{T_{\text{paralelo}}} = \frac{w_1 + w_2 + w_3 + w_4}{W_{\text{máx}} + h * G + M_s * L} = P, \tag{3}$$

lo cual representa una aproximación al *speedup* del modelo paralelo al trabajar con P procesadores. Haciendo referencia al procedimiento anterior:

$$v_1 = \frac{W_{\text{máx}}}{TP_{\text{máx}} + TB_{\text{máx}}}, \tag{4}$$

donde $W_{\text{máx}} = \text{máx}[TP + TB]$.

$$v_2 = \frac{TC + TS}{TC_{\text{máx}} + TS_{\text{máx}}} \tag{5}$$

$$v_3 = \frac{T_{\text{secuencial}}}{(W_{\text{máx}} + TC + TS) * P} \tag{6}$$

Aqui, v_1 representa la relación entre el tiempo de procesamiento mayor de un superstep y los máximos obtenidos de esos valores desde la base temporal determinada. v_2 representa la relación entre el tiempo

Figura 3: Paleta de Colores.

de Comunicación y Sincronización de un superstep y los máximos obtenidos de esos valores desde la base temporal determinada. v_3 representa la relación entre el tiempo secuencial y el tiempo paralelo de un superstep.

Con esta base, se realizó una serie de experimentos para evaluar el comportamiento de la herramienta, comenzando con una proporción alta de consultas simples. A continuación, luego de un cierto número de consultas se detecta el aumento de la proporción de consultas complejas con la consiguiente pérdida de performance de la Base de Datos. Cuando el administrador detecta dicha caída interviene ampliando la proporción de consultas simples, con lo que se restablece la performance anterior (representado por los colores neutros, gris en este caso, ver Fig. 2). Por otra parte, podemos observar en dicha figura la diferencia de performance de cada uno de los procesadores, por ejemplo el procesador 3 posee mayores capacidades de procesamiento, lo que se ve reflejado con colores más neutros (su TB es menor al de los otros tres procesadores).

La segunda metáfora visual explorada (ver Fig. 4) permite un análisis más minucioso de la historia del procesamiento de una serie de consultas. Ésta consiste en representar la performance de cada procesador en "cintas" verticales correspondientes a las variables del modelo de costos. A su vez cada cinta se encuentra dividida en carriles, representando a cada uno de los procesadores involucrados (P0, P1, P2 y P3 en nuestro caso). Cada bloque representa el valor de una variable en particular para un superstep o conjunto de supersteps dado. También en esta figura es posible observar el cambio en la performance de los procesadores a medida que va modificándose la proporción de consultas simples y complejas.

Para facilitar la visualización pueden agruparse varios de ellos en un solo bloque, disminuyendo la granularidad pero ofreciendo una imagen más integral sobre la performance de la base de datos distribuida (ver Fig.5). El tiempo está representado por la longitud de las cintas, donde el bloque inferior representa el conjunto de superstep más reciente. En la figura se puede observar cómo una visualización de menor granularidad permite observar detalles más ricos de la performance de los procesadores en instantes puntuales de la ejecución.

Por otra parte puede configurarse la paleta de colores para obtener representaciones más intuitivas. Por ejemplo, seleccionando una gama de crominancia que abarque desde el verde al rojo, donde rojo representa los valores de variables más elevados y verde representa los valores más bajos. Por último, puede configurarse la saturación, y luminancia, con las cuales podemos dar mayor énfasis a determinadas zonas de la paleta, donde se desea que el operador preste inmediatamente atención (ver Fig. 4) [29, 8].

Con los mismos datos que se utilizaron en la Fig. 4, en la Fig.5 pueden observarse claramente las diferencias entre los procesadores P0, P1 y P2 con respecto a P3, relacionados fundamentalmente con la performance de cada uno de los procesadores y los estados del servidor de procesamiento de consultas a lo largo de la ejecución del experimento como resultado del cambio de parámetros en el mismo. En los bloques más recientes se observa una baja performance de los procesadores P0, P1 y P2 en TB, como consecuencia de la definición de una alta proporción de consultas complejas en la cola de consultas. Consecuentemente P3, un procesador más potente, posee un TS mucho mayor, debido a que ha terminado sus tareas de procesamiento con anterioridad a los restantes procesadores. En este experimento se aumentó progresivamente la proporción de consultas complejas sobre las consultas simples.

7 Conclusiones

Un aspecto clave en la solución propuesta para el servidor es que éste esta construido sobre el modelo de computación paralela BSP. Es sabido que este modelo soporta una metodología estructurada de diseño de

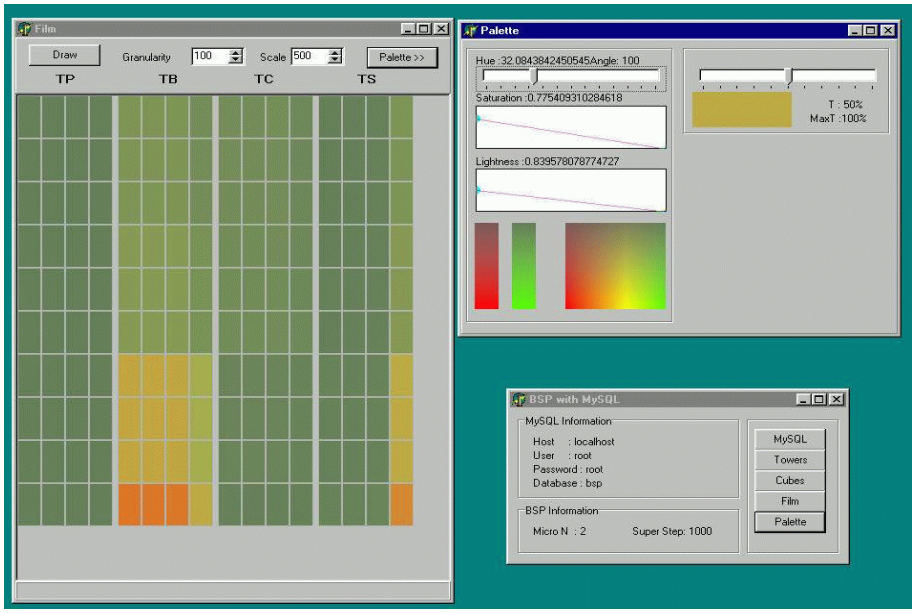


Figura 4: Sucesión de supersteps representada como línea de tiempo vertical.

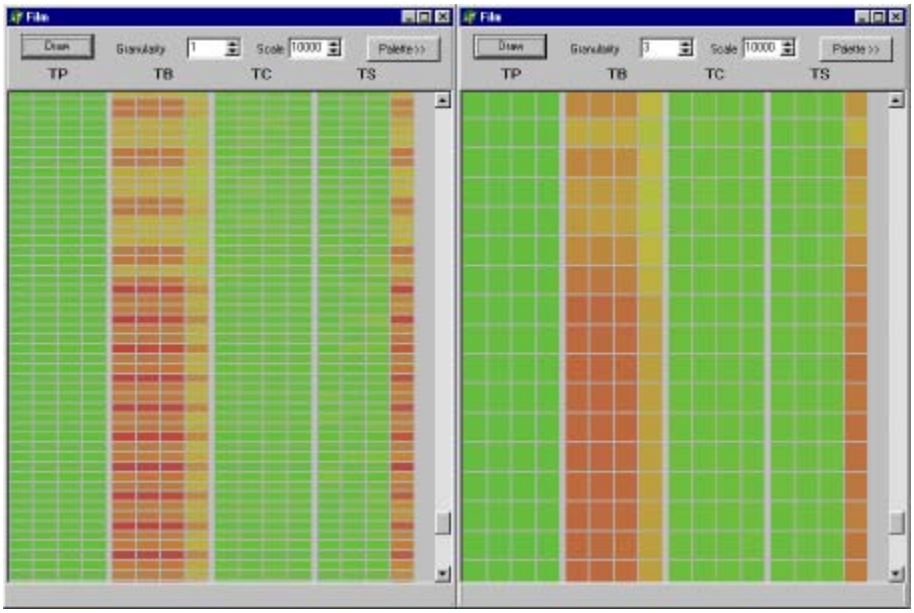


Figura 5: Diferentes granularidades en la visualización.

software que es simple de utilizar. Igualmente importante es el hecho de que la estructura del modelo permite cuantificar el costo de ejecución del software localizándose no solo en la componente de computación sino que también en las componentes de comunicación y sincronización. Existe una manera explicita de realizar esta cuantificación, la cual consiste en contabilizar la cantidad de computación y comunicación realizada en cada superstep, y luego sumar sobre todos los supersteps ejecutados.

La principal contribución de este trabajo está precisamente en la utilización del modelo BSP y su método de cuantificación de tiempo de ejecución para formular una herramienta gráfica que permite visualizar medidas de desempeño y en base a dicha visualización tomar decisiones respecto de la operación del servidor. En particular, en este trabajo nos hemos concentrado en el orden en que es conveniente procesar distintos tipos de consultas SQL que están arribando al servidor a una cierta tasa de llegadas. El objetivo es proporcionar al administrador de la base de datos una forma simple de visualizar lo que esta ocurriendo con el tráfico de consultas en un periodo dado de la operación del sistema con el fin de realizar optimizaciones.

Referencias

[1] ACM Computing Surveys. *Distributed query processing*, Dec. 1984.

[2] R. Bamford. Architecture of oracle parallel server. *VLDB'98*, pages 669–670, 1998.

[3] N. S. Barghouti and G. E. Kaiser. Concurrency control in advanced database applications. *ACM Computing Surveys*, pages 269–317, 1991.

[4] C. Baru, G. Fecteau, and A. Goya. Db2 parallel edition. *IBM Systems Journal*, pages 292–322, 1995.

[5] D. Bitton, H. Boral, D. J. DeWitt, and W. K. Wilkinson. Parallel algorithms for the execution of relational database operations. *ACM Transactions on Database Systems*, pages 324–353, Sep. 1983.

[6] G. Bozas, M. Jaedicke, and A. Listl. On transforming a sequential sql-dbms into a parallel one: First results and experiences of the midas project. *EuroPar'96*, pages 881–886, Aug. 1996.

[7] CACIC 2003. *Representación visual para la administración del procesamiento paralelo de consultas SQL*, La Plata - Argentina, Octubre 2003.

[8] C. Delrieux. Fundamentos e implementación de sistemas de computación gráfica. Departamento de Ingenieria Eléctrica, Universidad Nacional del Sur, 1999.

[9] D. DeWitt and J. Gray. Parallel database systems: The future of high performance database systems. *Comunications of the ACM*, pages 85–98, June. 1992.

[10] En Jornadas Chilenas de Computación 2001. *Procesamiento paralelo de consultas SQL generadas desde la Web*, Nov. 2001.

[11] B. Gerber. Informix on line xps. *ACM SIGMOD'95*, 24 of SIGMOD Records:463, May. 1995.

[12] G. Graefe. Query evaluation techniques for large databases. *ACM Computing Surveys*, page 73:170, June 1993.

[13] J.M.D. Hill, S. A. Jarvis, C. Siniolakis, and V. P. Vasilev. Portable and architecture independent parallel performance tuning using a call-graph profiling tool: A case study in optimising sql. Technical Report PRG-TR-17-97, Computing Laboratory, Oxford University, 1997.

[14] In 1997 International Conference on Parallel and Distributed Computing Systems. *Architectures for parallel query processing on networks of workstation.*, Oct. 1997.

[15] IX Jornadas Iberoamericanas de Informática. *Visualización Gráfica de Consultas SQL en Paralelo*, Cartagena-Colombia, Agosto 2003.

[16] M. Marín, J. Canumán, and D. Laguía. Un modelo de predicción de desempeño para bases de datos relacionales paralelas sobre bsp. *VI Congreso Argentino de Ciencia de la Computación*, Oct 2000.

[17] P. Mishra and M. Eich. Join processing in relational databases. *ACM Computing Surveys*, March 1992.

[18] C. Mohan, H. Pirahesh, W G. Tang, and Y. Wang. Parallelism in relational database management systems. *IBM Systems Journal*, pages 349–371, 1994.

[19] Oracle. *Oracle parallel server: Solutions for mission critical computing. Technical Report Oracle Corp.*, Feb. 1999.

[20] SC'97. *Parallel database processing on a 100 node pc cluster: Cases for decision support query processing and data mining*, 1997.

[21] D.B. Skillicorn, J.M.D. Hill, and W.F. McColl. Questions and answers about BSP. Technical Report PRG-TR-15-96, Computing Laboratory, Oxford University, 1996. Also in *Journal of Scientific Programming*, V.6 N.3, 1997.

[22] K. R. Sujithan. Scalable high-performance database servers. In *2nd IEE/BCS International Seminar on Client/Server Computing*, May 1997. IEE Press.

[23] K.R. Sujithan. Towards a scalable parallel object database — the bulk-synchronous parallel approach. Technical Report PRG-TR-17-96, Computing Laboratory, Oxford University, 1996.

[24] K.R. Sujithan and J. M. D. Hill. Collection types for database programming in the bsp model. In *5th EuroMicro Workshop on Parallel and Distributed Processing (PDP'97)*, Jan. 1997. (IEEE CS Press).

[25] URL. BSP and Worldwide Standard, http://www.bsp-worldwide.org/.

[26] URL. BSP PUB Library at Paderborn University, http://www.uni-paderborn.de/bsp.

[27] URL. MySQL Web Page, http://www.mysql.com/.

[28] L.G. Valiant. A bridging model for parallel computation. *Comm. ACM*, 33:103–111, Aug. 1990.

[29] Alan Watt. *3D Computer Graphics*. Addison - Wesley, 2da edition, 1993.

About the Performance of SQLf Evaluation Mechanisms

Yosmar López
Universidad Simón Bolívar, Departamento de Computación,
Apartado 89000, Caracas 1080-A, Venezuela
yosmara@hotmail.com

and

Leonid Tineo
Universidad Simón Bolívar, Departamento de Computación,
Apartado 89000, Caracas 1080-A, Venezuela
leonid@ldc.usb.ve

Abstract

In order to make more flexible database access the query language SQLf has been previously proposed. One of the SQLf features is the use of Fuzzy Quantifiers in Having Clause. For this kind of query, three evaluation mechanisms have been proposed: the Naïve, the Sugeno Integral Heuristics based and the Alfa-cut Derivation based. We present in this paper a formal performance study of these three mechanisms. This study has been made using a SQLf prototype build on top of a RDBMS.

Keywords: Database Fuzzy Querying, Query Performance.

1. INTRODUCTION

The problem of database fuzzy querying not is only a semantics issue but also a practical reality. The interest of some previous works has been to provide efficient evaluation mechanisms for fuzzy querying language SQLf [[1], [2], [3], [6]]. In case of fuzzy quantified queries, Bosc et al [[1]] have presented a strategy based in Sugeno Integral properties for improve query evaluation, we name it Sugeno Strategy. On the other hand, Tineo [[6]] has presented a strategy based in the distribution of the α-cut operator for evaluating fuzzy quantified queries, we name it Derivation Strategy. As ever, it is possible to apply an intuitive strategy pervaded of none improvement, we name it the Naïve Strategy. In this paper, we present the study of these strategies by mean of experimental proofs. We hope to determine which strategy gives the best performance and what are the conditions that ensure such behavior.

We present a system performance analysis that is based in experimentation and use of statistics models [[5]]. For so doing, we will make a design of experiments. As ever, the goal of this design will be to obtain the maxim of information with the minim quantity of experiments. The analysis of such experiments will lead us to distinguish the effects of factors that may inside in the system performance.

2. EVALUATION MECHANISMS

We consider the SQLf querying structure: *select t A from R group by A having Q are fc*. Being *t* a threshold associated with the query, *A* an attribute (attribute list) of *R* relation (relation list), *Q* a fuzzy quantifier, and *fc* a fuzzy condition. This query returns the fuzzy relation *Rf* on $\{a / (\exists x \in R / x.A=a) \wedge (\mu(Q(Xa,fc)) \geq t) \}$, being the membership degree of each element *a*: $\mu Rf(a) = \mu(Q(Xa,fc))$ (the truth degree of fuzzy quantified sentence *Q Xa's are fc)*, where $Xa = \{x \in R / x.A=a\}$. The sentence *Q Xa's are fc* is interpreted with the Yager's decomposition [[7]] interpretation. For example we may address a query to the employee relation of Table 1:

Table 1. Extension of EMP(#emp, e-name, salary, job, age, #dep)

#emp	e-name	Salary	Job	Age	#dep
10	Martin	2000	K1	40	1
22	Calvin	1000	K4	38	1
78	Luther	1500	K2	50	1
41	Johnson	1200	K3	40	2
35	Smith	1000	K3	39	2
90	Peters	1200	K2	41	2
56	Anderson	1500	K2	40	3
82	Dobson	1000	K4	36	3
64	Mc Dowell	2000	K1	50	3

If we want to f*ind the departments where most of the employees are about 40 years* we may use the SQLf query Ψ:

select 0.5 #dep from emp group by #dep having most_of age = about40.

Being *most_of* and *about40* the fuzzy defined in Fig. 1.

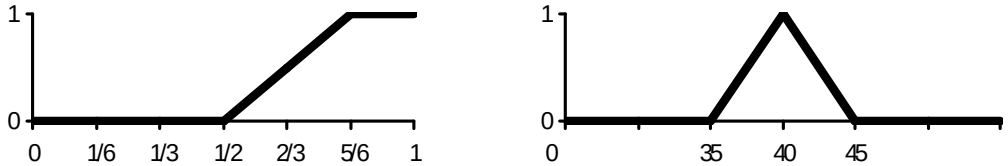


Fig. 1. Fuzzy Terms Membership Functions. At left, the proportional increasing quantifier *most_of*. At the right side, the fuzzy predicate *about40* that represents ages around 40 years old

According to the intended semantics of fuzzy quantified queries, we may compute the solution of the example query as Table 2 showns.

Table 2. Computation of Ψ query solution.

#dep	I	#emp	age	$\mu_i = {}^{i}\sup_{x \in Xa}(\mu_{fc}(x))$	$\mu_{Qi} = \mu_Q\left(\frac{i}{\lvert Xa \rvert}\right)$	$\min(\mu_{Qi}, \mu_i)$	$\mu\ (Q(Xa, fc))$
1	1	10	40	1	0	0	
	2	22	38	.6	.5	.5	
	3	78	50	0	1	0	.5
2	1	41	40	1	0	0	
	2	35	39	.8	.5	.5	
	3	90	41	.8	1	.8	.8
3	1	56	40	1	0	0	
	2	82	36	.2	.5	.2	
	3	64	50	0	1	0	.2

As the query specify an user desired threshold, the result set contains only those elements having satisfaction degree greater or equal to the threshold. We obtain finally Table 3.

Table 3. Result of Ψ query

#dep	Membership degree
1	.5
2	.8

The main idea of the studied mechanisms is to allow adding fuzzy querying capabilities on top of an existing RDBMS. Such mechanisms perform fuzzy query evaluation on the result of an underlying regular query addressed to the RDBMS. In this context a theoretical measure of mechanisms behavior is the number of accessed database rows. Anyone may think that whenever more rows are accessed more time is spent. Thus evaluation mechanisms have been proposed in order to keep low the number of accessed rows y query evaluation. Nevertheless, for so doing selection criteria may be of high complexity. Therefore it is necessary to perform an experimental study to show whether such mechanisms have or have not better performance than naïve solution.

Naïve Strategy[[1],[2]] consists in a program scanning the whole database relation and computing the satisfaction degrees. For previous example query, Naïve strategy will make all the computation shown in Table 2. We would like to avoid the whole database relation scanning. As ever, row access in expensive in time. Moreover, in fuzzy queries, satisfaction degree computation is also time consuming.

Sugeno Strategy[[1]] consists in scanning the whole table as in Naïve one, but with a halting conditions for each group. Conditions involve the minimum number of elements LB satisfying the fuzzy quantifier with a degree greater or equal than. In the example, for each group LB is N*2/3, being N the group cardinality. The failure condition when the number of group's scanned rows with satisfaction degrees under the desired threshold is greater than N-LB. In this case remaining rows in the group are ignored and the grouping attribute value is not part of the solution. In the example only the row of #emp=56 could be avoid in computation. We can see that this strategy avoids some rows access. The benefit of this technique is that it does not use any complex selection criteria. On the other hand the number of accessed rows depends of rows retrieving ordering. If fact, in worst case all rows are accessed despite the group doest not meet the threshold for satisfaction degree.

Derivation Strategy[[6]] consists in scanning the rows selected by a regular query intended for retrieving only those rows whose satisfaction degree is greater or equal to the satisfaction threshold and is member of a group containing at least LB such rows. This strategy avoids extra computation. In the example, only the rows of #emp=10, 22, 41, 35 and 90 are scanned. This strategy only accesses rows that are really relevant for satisfaction degree computation of groups that do meet the threshold. Regular query addressed to the RDBMS contains derived crisp selection criteria. The advantage of using such criteria is that we may think that RDBMS will make use of any optimization mechanism inside it. Nevertheless such criteria may be of high complexity. It may have a bad influence on total spent time.

Due to difference presents in these strategies for fuzzy query evaluation, it is an open issue their behavior in term of total spent time. As we have mentioned before, it is the main contribution of work presented in this paper.

3. EXPERIMENTS' DESIGN

The performance evaluation will be made using formal model statistic method. The idea of this method is to obtain a model that explains the influence of several considered factors in the observed values from experiments. For this kind of study we must establish the data that will be used for the experiment, the answer variables that will be measured and the factors that will be taken in account. Furthermore, we must propose the different experiments that will be performed. These experiments are determined by the different considered factors and the different levels for each factor. Finally, an initial model must be proposed in order to make the statistic analysis with the experimental results.

The queries will be addressed to a database relation. This relation will contain the experimental data for our study. Therefore we must define the scheme and the extension of this database relation.

We don't want to use a complex relation structure; rather we will propose a minimal structure. We will define the relation scheme with one attribute that may be used as primary key, another attribute susceptible of a fuzzy treatment and finally an attribute that may be used for grouping. With these criteria we define the relation *employee(identification_card,age,departament_code)*.

The relation extension is generated as follows: Values for identification_card are sequentially generated numbers. Values for age are uniform random generated numbers between 18 and 65. Values for deparment_code are uniform random generated numbers between 1 and 7.

The variables that are observed in the experiments are called answer variables. They usually are measures of the system behavior. In our case, we observe the time. The time of response refers to the total expend time of processing the fuzzy query. In other words is the time that the user waits for the complete system answer. This time is measured by the SQLf system prototype.

The number of possible experimental studies is infinite. Therefore we must fix some conditions in our study in order to limit it. However, we must ensure that the experimental study to be as general or representative as possible. In our case of study we will fix all query parameters except the fuzzy quantifier selectivity. We will use a query like: *select 0.5 departament_code from employee group by department_code having <fuzzy_quantifier> are age=Around40.*

We choose 0.5 as threshold. This chose allows us to take no care of calibration as a factor due to the following reasons: Fixed a value for the calibration, the selectivity of the query will be determined by the used fuzzy predicate and quantifier. The extreme values 0 and 1 lake of sense for selection, the level 0 is equivalent to do not establish a calibration, and the level 1 is equivalent to perform a regular query. The use of a middle value is imposed, we prefer 0.5 because it is the hope of a uniform variable into the unit interval.

In order to isolate the problem of strategy performance for quantifiers, we use a single fuzzy predicate. We also fix the fuzzy predicate as Around40 in Fig. 2.

Fuzzy quantifiers are classified in six categories. The classification obeys to the fuzzy quantifier's nature (absolute or proportional) and its membership function behavior (increasing, decreasing or unimodal). Nevertheless, fuzzy quantified sentences using any kind of quantifier may be transformed into sentences using only increasing proportional quantifiers. This argument has been used in previous works in order to simplify the study of fuzzy quantified sentences.

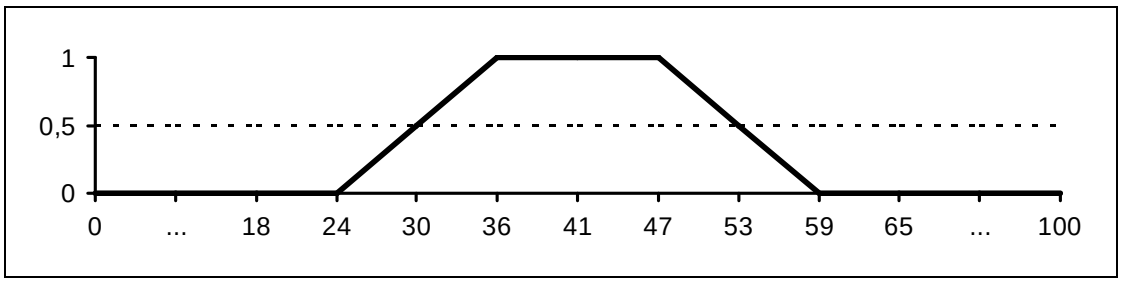


Fig. 2. Around40 Fuzzy Predicate. This user definition is not centered in 40. It was chosen for convenience of equal number of elements in the 0.5-cut and out of it for experimental data.

The experimental factors are those variables whose values are changed in the experiments in order to determine their effect in the answer variable. Factors may take different values. For the experiments' desig n it is necessary to choose some possible values than will be used for each factor. These chosen values are called the factors' levels. The combination of factors and levels used in the experiments will determine the used kind of design. Factors and levels of our study are described hereafter.

The objective of this study is to establish a comparison of the proposed strategies based in experimental results. We expect this factor to have a high influence in the performance of the query evaluation. Its levels are: Naive Strategy, Derivation Strategy and Sugeno Strategy.
As ever, the volume of data is a factor that must be considered. Despite in the different strategies the proportion of accessed registers does not depends on how large is the database, it is reasonable to think that an interaction might exists between the volume factor and the strategy factor. We define three levels for the volume factor. They are: low (1000 rows), middle (10000 rows), and high (100000 rows).
As we have said before, we have restricted our experiments to increasing proportional quantifiers. We define three quantifiers to be used in the experiments, they are represented in Fig. 3. These quantifiers will establish the three levels of the quantifier factor. We think that these three levels are representative of the different scenarios of selection imposed by the fuzzy quantifiers.

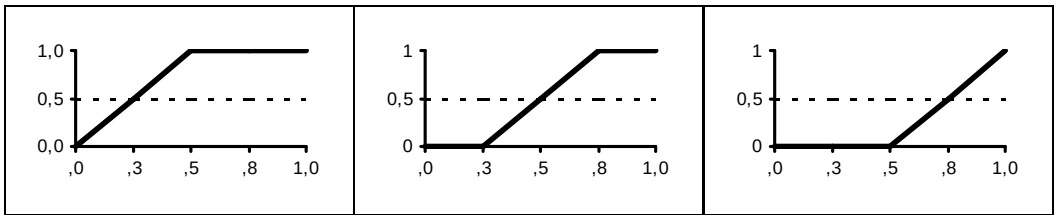


Fig. 3. Fuzzy Quantifiers Membership Functions. Left: *HalfOf* quantifier, the least selective. Center: *MostOf* quantifier, the middle in selectivity level. Right: *QuasiAll* Quantifier the most selective of considered quantifiers.

We have chosen a full factorial design for our experimental study. That is, we will consider all the mentioned factors and all their levels. This kind of design allows study the influence of each factor and all theirs interactions.

Table 4. Sumary of Factors. Each factor is denoted by a Symbol. Also shows factors' levels.

F	Symbol	Name	lf	Level 1	Level 2	Level 3
1	E	Strategy	3	Naïve	Derivation	Sugeno
2	V	Volume	3	Low	Middle	High
3	Q	Quantifier	3	Half Of	Most Of	Quasi All

The proposed experimental model for our study is:

$$y_{ijkl} = \begin{pmatrix} y_{\ldots} \\ + E_i + V_j + I_k + Q_l \\ + EV_{ij} + EI_{ik} + EQ_{il} + VI_{jk} + VQ_{jl} + IQ_{kl} \\ + EVI_{ijk} + EVQ_{ijl} + EIQ_{ikl} + VIQ_{jkl} \\ + EVIQ_{ijkl} \end{pmatrix}$$

Being:

- y_{ijk} the observed value for levels i, j, k of factors E, V and Q, respectively.
- $y_{\ldots}$ the arithmetic mean of the observed values of all experiments.
- F_m the effect of the factor F at the level m, $F \in \{E,V,Q\}$.
- $F'F''_{m'm''}$ the effect of the Interaction between factors F' and F'' at the levels m' and m'', respectively, with different $F', F'' \in \{E,V,Q\}$.
- EVQ_{ijk} the effect of the Interaction between all the factors E, V and Q for the levels i, j and k respectively.

4. Experimental Results

We have performed the experiments with the design presented in pervious chapter. The fuzzy queries where addressed to a SQLf prototype that we have developed [[4]] on top of Oracle 8i DBMS. This prototype allows the use of any of the three evaluation mechanisms. The prototype computes the total spent time for the evaluation of the

fuzzy query. We use (in dedicated mode) a SUN Enterprise 450 architecture server of two 250 MHz. processors with 512MB RAM, four 4GB SCSI hard disks and Solaris 8.
With the same environment conditions, we have run three times the experiments, obtaining similar results. We have made a test of standard deviation of these three replicas, showing that is not relevant to consider replicas in the model. Therefore, we only present here results of one replica. As resulting times difference is very high, we normalize them applying the logarithmic transformation: $y_{ijk} = \ln\left(\tau_{ijk} + 1\right)$, being τ_{ijk} the observed times.

Table 5. Logarithm Transformed Times. Experimental times have been normalized. We can see that transformation gives values of no more than 1 magnitude order of difference.

Strategy↓	Quantifier→ Volume↓	HalfOf	MostOf	QuasiAll
Naïve	Low	0.760806	0.732368	0.779325
	Middle	2.636912	2.597491	2.568788
	High	5.616662	5.550592	5.455107
Sugeno	Low	0.792993	0.625938	0.518794
	Middle	2.614472	2.618855	2.080691
	High	5.782224	5.719295	5.141488
Derivation	Low	0.431782	0.285179	0
	Middle	2.102914	1.759581	0.009950
	High	5.355642	4.011506	0

We use R Statistical Software for the analysis of variance. We introduce in this tool the normalized experimental. Thereafter, we specify the model and inspect it.
The analysis is made with the statistical F distribution proof. We may observe in Table 6 (ANOVA Table) that: All factors, Strategy, Volume and Quantifier and their interactions have a high influence in the behavior of the observed value. In these cases the computed F values are very close to the tabled F values (they are '***' marked). It tells us that we may conclude about the answer variable behavior respects to the factors and its interactions, with statistics certainty.

Table 6. Analysis of Variance Full Factorial Model. Asterisk marks in rows denote the relevance of the factor or factor interaction in explaining the experimental results.

	Df	Sum Sq	Mean Sq	F value		Pr(>F)	
Strategy	2	33.946	16.973	1,32E+36	<	2.2e-16	***
Volume	2	242.048	121.024	9,44E+36	<	2.2e-16	***
Quantifier	2	16.645	8.322	6,49E+35	<	2.2e-16	***
Strategy:Volume	4	11.765	2.941	2,29E+35	<	2.2e-16	***
Strategy:Quantifier	4	18.601	4.650	3,63E+35	<	2.2e-16	***
Volume:Quantifier	4	8.528	2.132	1,66E+35	<	2.2e-16	***
Strategy:Volume:Quantifier	8	12.145	1.518	1,18E+35	<	2.2e-16	***
Residuals	54	6,93E-27	1,28E-28				

Signif. codes: 0 ` ***' 0.001 ` **' 0.01 ` *' 0.05 `.' 1							

In order to show the influence of factors, we plot the observed results as functions of each couple of relevant factors. Remember that we have made a logarithmic transformation of the observed data. The graphics forms will lead us to understand the influence of factors. We present only interaction with Strategy factor because it is the main interest of this study.
Influence in response time of Strategy and Volume interaction plot is shown in Fig. 4- We may observe there that the behavior of times for any strategy is increasing respect to the growth of the data volume. This result is not surprising at all; we expected this influence of the volume factor as ever. The volume factor is very important in the explanation of the studied performance.
On the other hand, we may remark the quasi-equal graphics of Naïve and Sugeno strategies; it is little the benefit observed of Sugeno strategy respect the Naive one. Nevertheless, we may note a high benefit of using the Derivation

strategy respect the other ones; the values for the Derivation strategy are lower than values for either Sugeno or Naive strategies. This tells us that the strategy factor is definitively influencing the performance of query evaluation. Moreover, this leads us to affirm that the Derivation strategy ensures the better performance for the query evaluation.

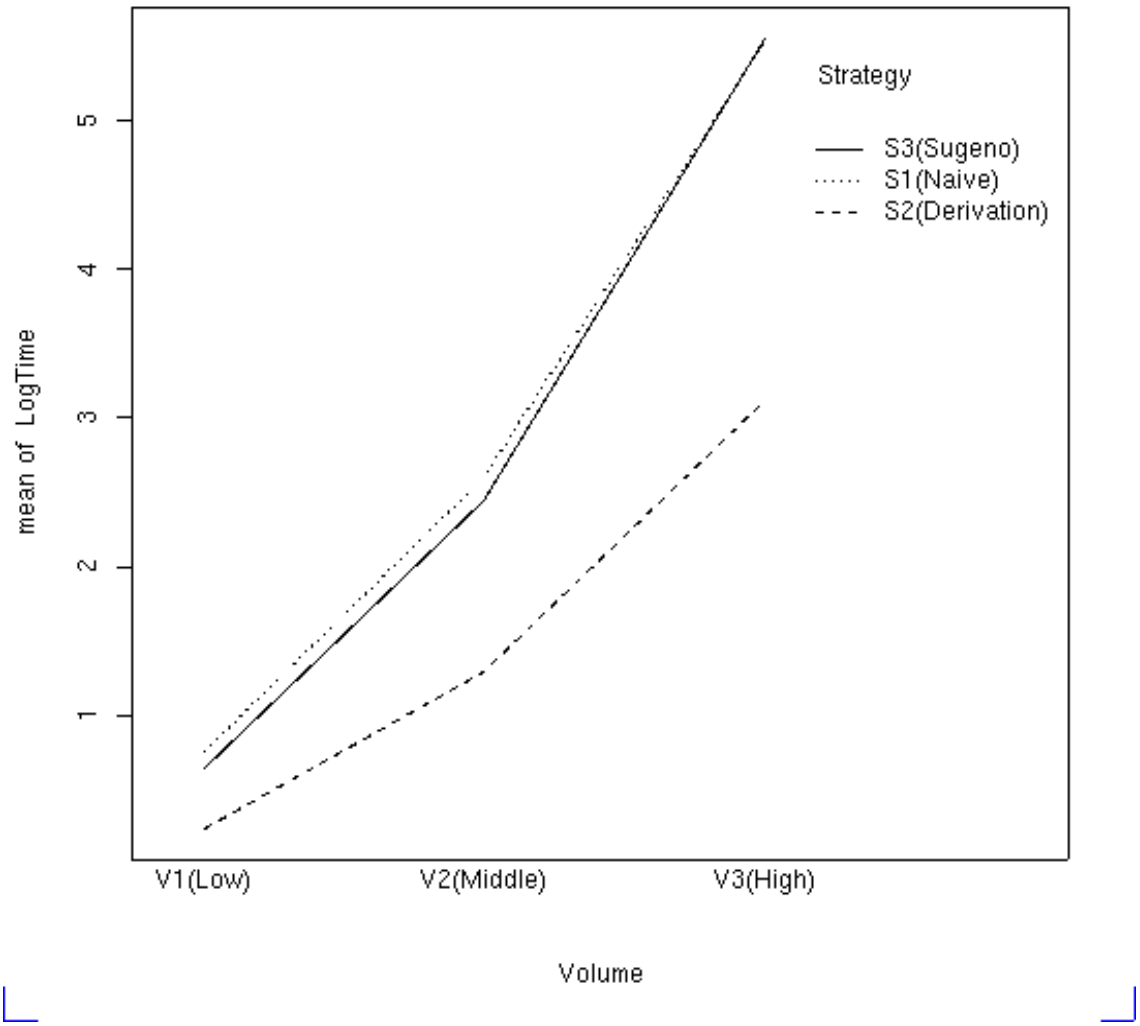


Fig. 4. Influence in Response Time of Strategy and Volume Interaction.

In Fig. 5 plot, corresponding to influence of Strategy and Quantifier interaction, once again we observe that the Derivation strategy presents better performance than Sugeno and Naive ones. In this interaction the behavior of the Sugeno strategy is different to the behavior of the naive Strategy. In case of a quantifier that imposes a more strict selection condition, Sugeno strategy has better performance than Naive one. The mean time for the Naive strategy stays constant no matter the used quantifier. It obeys to the fact that the Naive strategy takes no advantage from the fuzzy selection conditions. There is a clear influence of the Quantifier in answer time, it is evidenced by the graphics for the Sugeno and the Derivation strategies. Interaction of Strategy and Quantifier has the higher significance in explaining query spent time according to Analysis of Variance (Table 6).

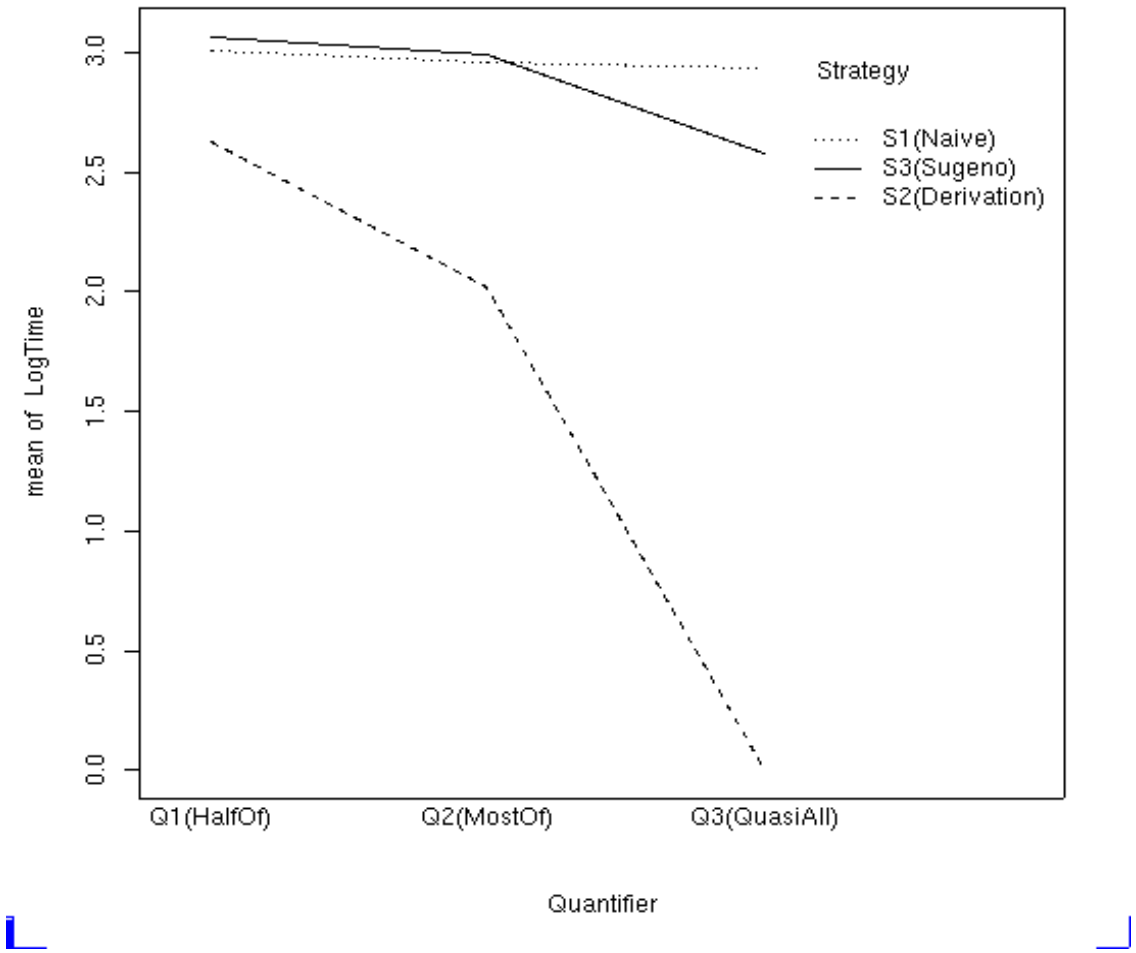


Fig. 5. Influence in Response Time of Strategy and Quantifier Interaction.

5. Concluding Remarks

We have made formal performance study of query evaluation mechanisms: Naïve, Sugeno and Derivation. In the study we have observed as answer variable the total spent time of the evaluation algorithm. This time is given by a SQLf prototype.

We have limited the study to partitioned queries. This kind of query is representative enough for all fuzzy quantified queries because any query involving a fuzzy quantifier may be transformed into this structure. We choose the level 0.5 as threshold for the calibration of the answers. Fixed a value for the calibration, the selectivity of the query will be determined by the used fuzzy predicate and quantifier. We prefer 0.5 because it is the hope of a uniform variable into the unit interval. We have also fixed the predicate under the fuzzy quantifier scope as a single predicate defined by a trapezium. Fuzzy quantified sentences using any kind of quantifier may be transformed into sentences using only increasing proportional quantifiers. Therefore we restrict the study to this kind of quantifiers.

We have chosen a full factorial design with the factors: E: Strategy (Naive, Derivation, Sugeno); V: Volume (Low, Middle, High); and Q: Quantifier (HalfOf, MostOf, QuasiAll). The analysis leads us to conclude that all these factors and interaction are very significant in explaining observed times.

As ever, the factor of higher influence in the performance is the Volume of data. The factor with a second high influence is the Strategy. It confirms the great importance of the strategy choice in query evaluation.

We must remark it is little the benefit observed of Sugeno strategy respect the Naive one. Nevertheless, we may note a high benefit of using the Derivation strategy respect the other ones. The relevance of this factor and its interactions with others considered factors leads us to conclude that the use of the Derivation strategy guaranties the best performance.
We work now in proposing new evaluation methods for fuzzy quantified queries on top of a RDBMS. We are also performing tests as that presented here for others fuzzy quantified querying structures. In further works it is possible to study the problem of fuzzy querying with acceleration structures such as indexes and also combine these evaluation methods with regular query optimization techniques. Another topic of further works interest is the problem of the interpretation of sentences of form *Q B X's are A*, its application to database querying in the context of SQLf, the evaluation mechanisms for queries involving this kind of sentences and the performance study of such mechanisms

References

[1] Bosc P., Liétard L., Pivert O., "Evaluation of Flexible Queries: The Quantified Statement Case", Proceedings of the 8th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems IPMU'2000, Madrid, España, (2000), Pp. 1115-1122.

[2] Bosc P., Pivert O., "SQLf query functionality on top of a regular relational DBMS", in: Knowledge Management in Fuzzy Databases, O. Pons, M.A. Vila, and J. Kacprzyk (Eds.), Heidelberg: Physica-Verlag, to appear.

[3] Liétard L., "Contribution à l' interrogation Flexible de Bases de Données: étude des propositions quantifiées floues" Thèse de Docteur Université de Rennes. (1995)

[4] López Y., "Evaluación de Estrategias para Consultas Cuantificadas en SQLf", Informe final de proyecto de grado en la Universidad Simón Bolívar, abril 2002.

[5] Raj J., "The Art of Computer Systems Performance", John Wiley/Sons, Inc, 1991.

[6] Tineo L., "Extending the power of RDBMS for Allowing Fuzzy Quantified Queries". Lecture Notes in Computer Sciences, September 2000. Vol. 1873, pp 407-416.

[7] Yager, R., Interpreting Linguistically Quantified Propositions, International Journal of Intelligent Systems, Vol. 9, (1994), Pp. 541-569.

Estimador de Tamaño de Colpas en Molienda Semiautógena Utilizando Horizonte Móvil Neuronal

Karina Carvajal, Gonzalo Acuña, Francisco Cubillos, Luis Magne
Universidad de Santiago de Chile, Facultad de Ingeniería, Santiago, Chile
{klcarvajal;gacuna}@diinf.usach.cl

Abstract

The development of a moving horizon estimator coupled with an external neural network acting as a dynamic model of a semiautogenous grinding process is presented. A phenomenological model describing the evolution of size particles was simulated using Matlab Simulink in order to have enough data for training the neural network and validate the estimator performance. A gaussian noise was added to the variables in order to better simulate real conditions. Good results for the estimation of the relevant variables are shown (adecuation index over 0,9).

Keywords: Dynamic Neural Networks, Moving Horizon State Estimator, Local Optimisation , Semiautogenous Grinding

Resumen

En el presente trabajo se presenta el desarrollo de un estimador de horizonte móvil acoplado a una red neuronal de recurrencia externa que hace las veces de modelo dinámico de un proceso de proceso de molienda semiautógena. Para el entrenamiento de la red neuronal y posterior utilización del estimador se hizo uso de datos simulados por un programa realizado con la herramienta *Simulink* de Matlab confeccionado sobre la base de un modelo fenomenológico existente que describe el proceso de fractura de las colpas de mineral en dichos molinos. Las variables fueron contaminadas con ruido aditivo gaussiano para simular mejor el comportamiento real del proceso. Los resultados de la estimación de las variables relevantes (tamaño de colpas) son satisfactorios con índices de adecuación superiores a 0,9.

Palabras claves: Redes Neuronales Dinámicas, Estimación de Horizonte Móvil, Optimización Local, Molienda Semiautógena

1 Introducción

Los sensores virtuales (software-sensors) han probado ser una herramienta poderosa en la determinación de variables de estado no medibles [10]. En general para este tipo de sensores es necesario contar con modelos dinámicos apropiados, que den cuenta de la evolución de las variables de estado relevantes. Muchas de las técnicas utilizadas en la implementación de sensores requieren modelos descriptivos bastante precisos del proceso, los que, para el caso industrial, son difíciles de obtener. Adicionalmente ellas son exigentes en tiempo de desarrollo, delicadas en cuanto a su calibración y puesta a punto y no consideran, en muchas ocasiones, modelos realistas de las perturbaciones, propias de un medio industrial. Una buena alternativa para desarrollar modelos a partir de la información de entrada-salida del proceso son las redes neuronales, entendidas como aproximadores universales de funciones no-lineales [6].

El estimador de estado de horizonte móvil (MHSE-del inglés Moving Horizon State Estimation) es un enfoque en donde se transforma un problema de estimación dinámica en otro de optimización no-lineal sobre un horizonte temporal y se presenta como una alternativa para desarrollar técnicas de estimación que cumpla con las condiciones en los procesos a escala industrial. Un aspecto central en este método lo constituye la minimización de una función objetivo o criterio consistente en la suma de los errores de predicción de la salida sobre un cierto horizonte de tiempo. Es decir, MHSE busca el vector de estado, al inicio del horizonte, que minimiza el criterio escogido.

En molienda semiautógena (SAG), ha habido esfuerzos previos por desarrollar un modelo fenomenológico de balance de masas [8]. Este modelo explica el transporte de masa a escala piloto dentro de un molino semiautógeno. Aunque los resultados obtenidos son buenos, su complejidad hace difícil llevarlo a escala industrial a fin de abordar problemas de optimización y control. Por lo que se propone el desarrollo de un estimador correspondiente a un Estimador de Horizonte Móvil, que ha probado ser muy efectivo y simple de operar en procesos industriales [11]. En esta ocasión el modelo dinámico que sirve para el funcionamiento del estimador es una red neuronal de recurrencia externa, la que permite aproximar la evolución de ciertas variables de estado del proceso dinámico no-lineal de molienda SAG. Se aprovecha así la capacidad de las redes neuronales de aproximar funciones no lineales complejas con una precisión arbitraria [4]. A partir del modelo fenomenológico apropiadamente implementado utilizando el software Simulink de Matlab se generaron datos que sirvieron para el entrenamiento de la red neuronal.

Este trabajo se divide en las siguientes secciones: una breve descripción del proceso de molienda SAG, el modelo neuronal utilizado, una definición de lo que es un estimador de horizonte móvil, la metodología empleada junto a los resultados obtenidos y las conclusiones finales.

2 Molienda Semiautógena

El objetivo de los Procesos de Concentración en minería es recuperar las partículas de especies valiosas (cobre, oro, plata, etc.), que se encuentran en las rocas mineralizadas. El proceso de concentración se divide en tres fases: Chancado, Molienda y Flotación .
En la molienda autógena actual (AG), rocas de hasta 8 pulgadas o más son alimentadas a un molino cilíndrico, cuyo diámetro es 2 a 3 veces su largo. La palabra autógena indica que la molienda ocurre debido a la propia acción de caída de las colpas minerales desde una altura cercana al diámetro del molino. La molienda semiautógena es una variación del proceso de molienda autógena y en ella se adicionan medios de molienda metálicos al molino, cuyo volumétrico de llenado varía de 4 a 14% del volumen del molino, como se muestra en la figura 1.

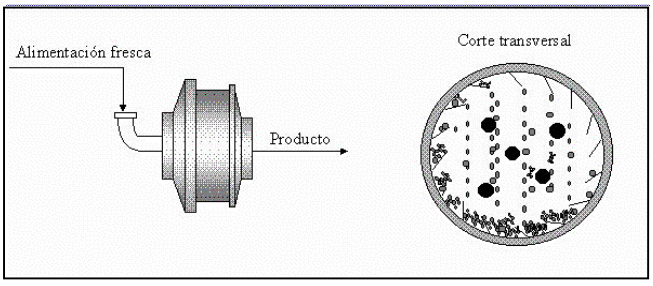


Figura 1. Esquema del molino SAG – Vista externa e interna.

En la operación de estos molinos se busca trabajar en condiciones que impliquen el máximo consumo de potencia instalada. Pero esto implica trabajar en un punto de operación inestable ya que un aumento en el nivel de llenado del molino más allá del punto de consumo máximo conduce a una condición de sobrellenado de éste (ver figura 2).

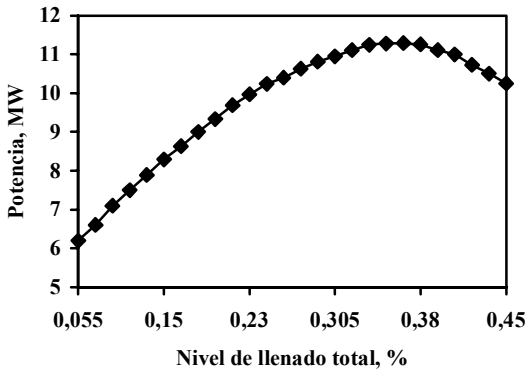


Figura 2. Variación típica del consumo de potencia con el nivel volumétrico de llenado total del molino.

Además, debe tenerse en cuenta que el valor de potencia máxima que puede consumir un molino semiautógeno no es constante y depende principalmente de la densidad de la carga interna, la distribución de tamaños de la alimentación y del estado del revestimiento. El nivel de llenado de la carga interna que corresponde al máximo consumo de potencia está relacionado con el nivel de llenado de medios de molienda y el movimiento de la carga interna. Por ello, los operadores de circuito de molienda semiautógena deben intentar conjugar estos factores para lograr, primero estabilizar la operación y posteriormente buscar su mejoramiento.

Para la operación estable de un molino semiautógeno se requiere tres condiciones: una adecuada proporción de las fracturas gruesas, intermedias y fina en la alimentación fresca, que permita al molino reponer los medios moledores; un flujo de alimentación fresca al molino que permita balancear la tasa de ingreso del mineral grueso con su tasa de molienda hacia tamaños más pequeños; y una tasa de descarga a través de la parrilla del molino que permita evacuar el mineral fino a la misma tasa que ingresa y que se genera por la fractura de los tamaños superiores [9].

Para modelar la molienda SAG y dadas las características físicas de este tipo de equipos, el molino se ha dividido en las siguientes partes.

a. Cámara de molienda: Aquí, se identifica y modela la ocurrencia del proceso de molienda y transporte de masa.

b. Cámara de descarga: Aquí, se verifica la clasificación por la parrilla interna y la posterior evacuación del material como producto final mediante los alzadores de pulpa.

Las partículas alimentadas al molino ingresan a la cámara de molienda, cuyo producto enfrenta a la parrilla interna, donde, de acuerdo a una probabilidad de clasificación, las partículas pueden permanecer en la cámara de molienda o formar parte del flujo de descarga, ya sea que éstas hayan o no atravesado la parrilla [7].

En la figura 3 se muestra la representación esquemática del molino semiautógeno en donde:

(1) Representa el Molino Real.

(2) Representa la Cámara de molienda.

(3) Representa la Parrilla interna.

Y F y P : son la alimentación y el producto del molino.

F* y P* : son los flujos de alimentación y producto del molino imaginario interno.

T : es el flujo de recirculación.

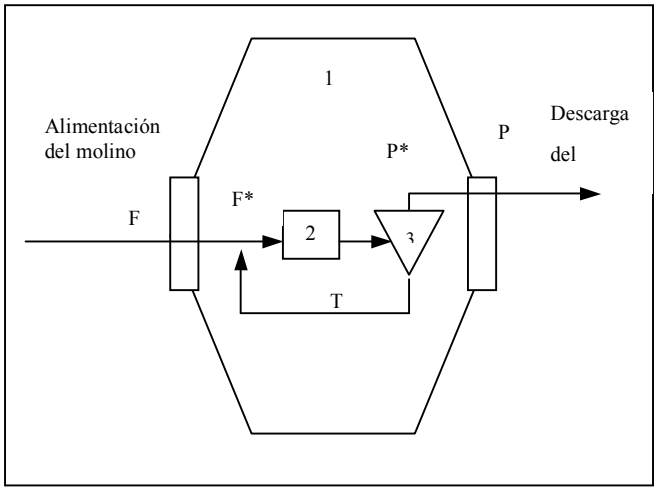


Figura 3. Representación esquemática de modelo de molienda semiautógena.

En base a este enfoque se describe el modelo fenomenológico para el molino semiautógeno, el cual plantea una ecuación diferencial respecto de la variable peso (ω), obteniéndose un modelo para la dinámica de cada uno de los tamaños de partículas determinadas dentro del molino [8], el cual se muestra en la ecuación (1).

$$\frac{d\omega_i}{dt} = -\left(\frac{P^*}{W}\right)(1-c_i)\omega_i - K_i * \omega_i - (K_{i-1} - K_i) * \sum_{k=1}^{i-1} \omega_k + f_i \qquad (1)$$

Donde : ω_i : Carga interna en *ton*.

f_i : Flujo de alimentación en *ton/h*.

K_i : Velocidad efectiva de molienda.

c_i : Factor de eficiencia de clasificación.

W : Peso total del mineral en la carga interna.

El modelo anterior describe una ecuación parcial que se encuentra completamente definida con los datos de entrada y de realimentación de los pesos internos del mineral dentro del molino.

3 Modelamiento mediante Redes Neuronales

A partir del modelo fenomenológico de 'balance de masas', apropiadamente implementado utilizando el software Simulink de Matlab, se generaron datos que sirvieron para el entrenamiento de la red neuronal. Se añadió ruido gaussiano a las salidas de amplitud igual a un 5% de la amplitud de la señal sin ruido. Como señales de entradas se utilizaron señales binarias pseudoaleatorias que simulan la cantidad de mineral que ingresa al molino. Las salidas corresponden al mineral que es evacuado del molino (ver figura 3) [5].

El criterio utilizado para formar grupos de datos está referido a los tramos lineales que se encuentran en la curva de eficiencia de clasificación resultando cinco grupos correspondientes a distintos tamaños en la alimentación del molino, los cuales son: colpas de mineral muy grande, grandes, medianas, pequeñas y colpas de mineral muy pequeñas. El mineral evacuado del molino corresponde a los tres tamaños inferiores descritos anteriormente, es decir, colpas medianas, pequeñas y muy pequeñas.

La arquitectura del modelo neuronal mostrada en la figura 4 contempla como variables de entrada a los cinco tamaños diferentes de la alimentación del molino (variables de control) y como variables de estado a los tres tamaños menores, pues se supone que al entrar al molino, y luego del proceso, el mineral más grande se muele y sólo se obtiene el mineral que tiene un diámetro menor a la parrilla de descarga.

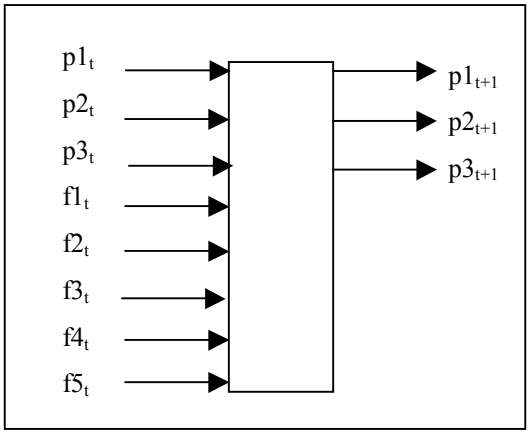


Figura 4. Arquitectura de la red.

Inicialmente se diseñó una red multicapa con recurrencia externa, a la cual se le fueron modificando los parámetros para obtener mejores resultados. Se trabajó modificando la cantidad de capas ocultas (1 y 2 capas), neuronas por capas (entre 5 y 20 neuronas en la capa oculta), funciones de transferencia, tipo y tamaño de señales de entrada y algoritmos de entrenamiento.

Los resultados de los parámetros que simulaban de mejor manera el proceso son:

Número de capas ocultas	= 1
Neuronas capa oculta	= 10
Función de transferencia	= logarítmica sigmoide en ambas capas
Algoritmo de entrenamiento	= algoritmo de retropopagación quasi-Newton (trainbfg)
Número de épocas	= 1000
Número de entradas	= 8

De 1000 datos aproximadamente generados por el modelo fenomenológico, del tipo señales binarias pseudoaleatorias, se consideraron 500 datos para entrenamiento y el resto para validar el modelo. En la prueba de la red se utilizaron señales sinusoidales, las cuales se ocuparon para comprobar la buena generalización que estaba realizando ésta. A las entradas ocupadas tanto en el entrenamiento y en las pruebas se les incorporó ruido.

4 Estimador de Horizonte Móvil

En general los estimadores clásicos suponen un modelo preciso del sistema, supuesto que, al no poder ser satisfecho en la mayor parte de las aplicaciones reales, se traduce en un error de la estimación que suele incrementarse con el tiempo. Adicionalmente y debido a la dificultad de incorporar en los métodos anteriores algunas restricciones típicas de ambientes industriales, como muestreo poco frecuente, respeto a leyes de conservación e inigualdades y necesidad de recalibrar cada cierto tiempo la estimación a partir de mediciones adecuadas, es que surgió una técnica de estimación adaptada de conceptos provenientes de métodos de control predictivo. Es el llamado Estimador de Horizonte Móvil, MHSE [1].

En síntesis, MHSE transforma un problema de estimación dinámica de estados en un problema de optimización estática, en el cual se busca el valor de un vector inicial al inicio de un horizonte de manera de minimizar la diferencia entre la salida real del sistema y la evolución de él dada por el modelo y sus condiciones iniciales. Este principio es ilustrado en la figura siguiente (Figura 5).

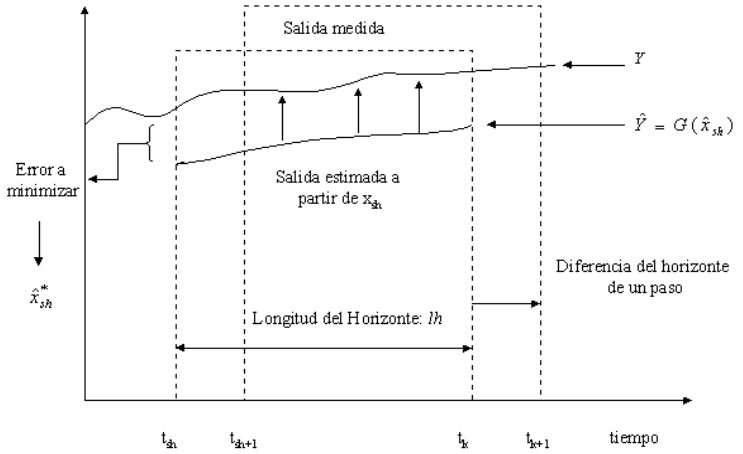


Figura 5. Representación esquemática del principio de estimación utilizado por MHSE.

Un aspecto central en este método lo constituye la minimización de una función objetivo o criterio consistente en la suma de los errores de predicción de la salida sobre un cierto horizonte de tiempo. Es decir, MHSE busca el vector de estado, al inicio del horizonte, que minimiza el criterio escogido.

Formalmente, si se tiene el modelo no lineal de un sistema dado por:

$$\dot{x}(t) = f\big(x(t), u(t)\big) \tag{2}$$

$$y(t) = g(x(t)) \tag{3}$$

MHSE busca determinar el vector, al inicio del horizonte, que minimice un criterio, por ejemplo del tipo:

$$J = \frac{1}{2}\sum_{i=t_{sh}}^{t_k}\left[Y_i - \hat{Y}_i\right]^2 \tag{4}$$

Siendo $Y = \left[y_{sh}, \ldots, y_k\right]^T$ la salida sobre el horizonte considerado, e $\hat{Y} = \left[\hat{y}_{sh}, \ldots, \hat{y}_k\right]^T$ las estimaciones de la salida sobre dicho horizonte.

El algoritmo MHSE puede resumirse en las siguientes sentencias:

1. Inicialización del vector de estados : $\hat{x}_{sh}$
2. Búsqueda del vector de estados inicial óptimo: $\hat{x}^*_{sh}$ (mediante algún método de optimización)
3. Cálculo de la solución al final del horizonte (instante actual t_k): $\hat{x}_k$
4. Retorno a la etapa 2 para efectuar el cálculo de $\hat{x}_{k+1}$, desplazando el horizonte en un paso.

En el caso de sistemas complejos no-lineales el problema de la optimización para determinar el vector de estados óptimo al inicio del horizonte no puede ser resuelto de manera analítica y es por esta razón que se debe utilizar técnicas numéricas. Estas pueden agruparse, *grosso modo* en técnicas clásicas o heurísticas, dentro de las que se puede destacar aquellas local o globalmente convergentes [2]. En este trabajo se utilizó una técnica local de optimización.

5 Estimador de Horizonte Móvil Neuronal

Para desarrollar un estimador de horizonte móvil neuronal, en vez de utilizar el modelo fenomenológico del proceso, se utilizó la red neuronal previamente entrenada como modelo dinámico del proceso (figura 4) en la cual se realimenta el valor de las variables de estado (figura 6). Esta red permite entonces estimar el valor de

dichas variables de estado al final del horizonte, a partir del valor óptimo inicial de ellas, otorgado por el método de horizonte móvil.

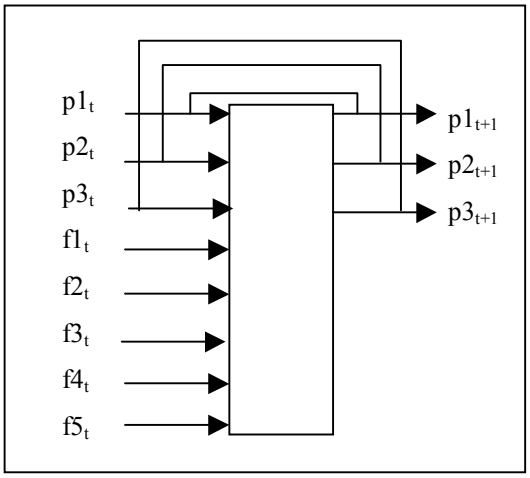


Figura 6. Esquema neuronal en la predicción de las variables finos y nivel de llenado.

A continuación se muestra la metodología utilizada en el desarrollo del estimador y los resultados obtenidos.

5.1 Descripción de la Metodología

El algoritmo de horizonte móvil utilizado es el siguiente:

1.- Inicializar: MaxT, sh, lh

2.- Mientras (sh+lh) ≤ MaxT

** Buscar el vector de estado inicial óptimo utilizando optimización local.*

**Para cada punto a lo largo del horizonte*

Obtener las variables de estado utilizando el modelo neuronal de la figura 6

Avanzar en un paso.

FinPara

**Almacenar los estados estimados.*

**Obtener las variables de estado utilizando el modelo neuronal de la figura6 para buscar el próximo estado óptimo.*

sh=sh+1

FinMientras

El método de optimización utilizado es el incluido en la función fmincon de Matlab, el cual encuentra el valor mínimo de una función no lineal multivariable, sujeta a restricciones.

La función a minimizar es la raíz cuadrada de la sumatoria de las diferencias entre los valores de colpas medianas y pequeñas predichos por la red neuronal y los valores deseados o experimentales. El método arroja los valores iniciales de las variables de estado que minimizan dicha función.

5.2 Resultados Obtenidos

Para evaluar el funcionamiento del estimador se utilizaron datos del conjunto de validación, correspondientes a aproximadamente 400 datos de señales sinusoidales. Las salidas referentes a la estimación y al valor deseado son comparadas utilizando los índices de error correspondientes al Error Cuadrático Medio (RMS), el Error Residual Estándar (RSD) y el Índice de Adecuación (IA) [3].

La figuras 7, 8 y 9 muestran los resultados obtenidos por el estimador y las salidas deseadas para las variables de salida colpas medianas, pequeñas y muy pequeñas, respectivamente. El horizonte utilizado por el estimador es de 30 unidades de tiempo.

La tabla 1 muestra el resultado de los índices obtenidos por medio de las pruebas del método de estimación de horizonte móvil neuronal.

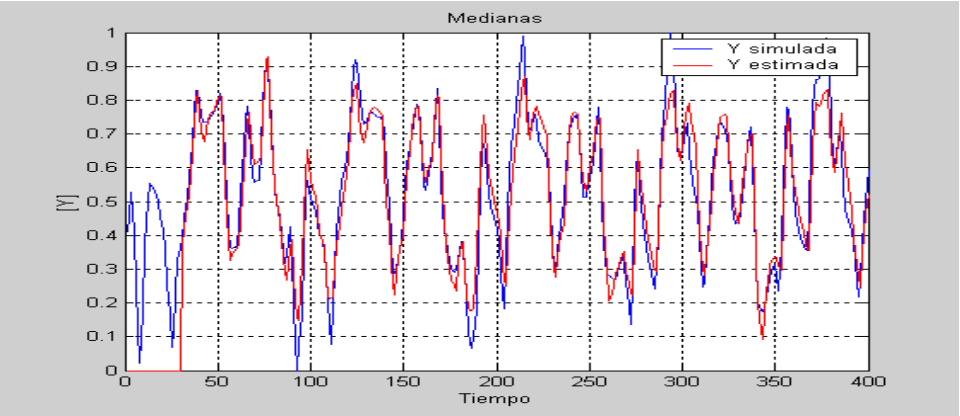


Figura 7. Salidas deseada y reales del estimador con respecto a las colpas medianas.

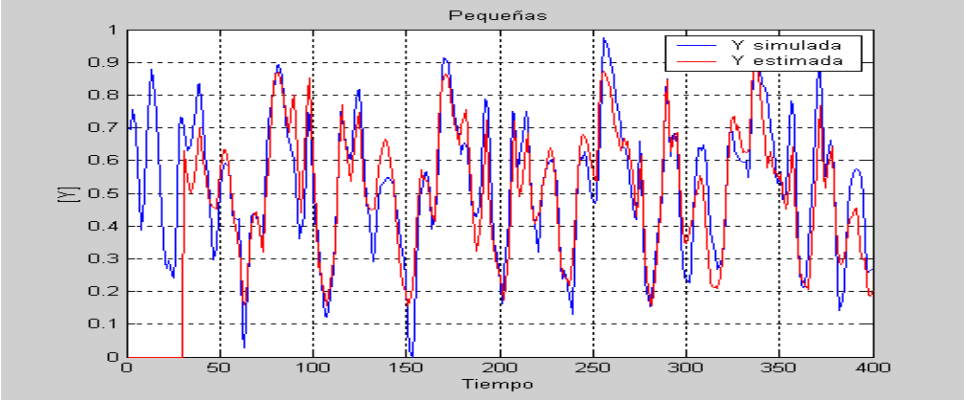


Figura 8. Salidas deseada y reales del estimador con respecto a las colpas pequeñas.

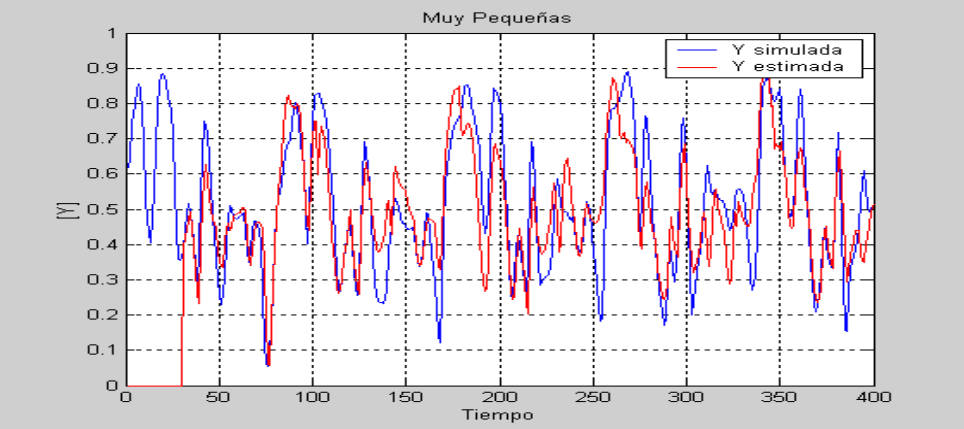


Figura 9. Salidas deseada y reales del estimador con respecto a las colpas muy pequeñas.

Tabla 1. Resumen de los índices de error del modelo

	Colpas medianas	Colpas pequeñas	Colpas Muy Pequeñas
IA	0.9766	0.9440	0.9018
RMS	0.1052	0.1584	0.1969
RSD	0.0612	0.0899	0.1057

De los resultados expuestos anteriormente se puede apreciar que el estimador logra predecir en un 98% las colpas medianas, en un 94% las colpas pequeñas y en un 90% las colpas muy pequeñas, que es la variable de estado que se desea estimar en línea y tiempo real. Resultados que muestran una buena estimación y que mejoran a medida que se aumenta el horizonte.

6 Conclusiones

En este trabajo se presenta un Estimador de Estados de Horizonte Móvil Neuronal, aplicado a un proceso de molienda semiautógena, el que a través de una optimización local busca valores óptimos de las variables colpas medianas, pequeñas y muy pequeñas a través de una red neuronal durante un horizonte de tiempo determinado.

En el modelamiento del proceso se utilizó una red estática con recurrencias externas, lo que permite modelar un sistema dinámico. El algoritmo de entrenamiento utilizado es el clásico de retropropagación del gradiente con él método de minimización de segundo orden quasi-Newton y función de transferencia logarítmica sigmoide, requiriéndose de los datos en un tiempo anterior t para predecir la salida en un tiempo t+1. El modelo neuronal dinámico obtenido permite la evolución del sistema sólo a partir de condiciones iniciales.

Los resultados entregados por el estimador, son muy satisfactorios dados el índice de adecuación (ia) mayor a 0.9 y el error cuadrático menor a 0.1 obtenidos, lo que indica una predicción muy cercana al valor real.

Debe destacarse que en este trabajo se demuestra la posibilidad de complementar el método de Horizonte Móvil, de creciente utilización en procesos industriales, con las redes neuronales, establecidas como modelos dinámicos, lo que permite potenciar las ventajas de ambas metodologías en una aplicación de suma importancia económica en nuestro país.

Agradecimientos

Se agradece financiamiento parcial de proyectos FONDEF D02I1077 y FONDECYT 1040208, Chile

Referencias

[1] Allgöwer F., Badgwell, T., Qin, J., Rawlings, J, Wright S. Nonlinear predictive control and moving horizon estimation – an introductory overview, in Advances in Control Highlights of ECC'99, Paul M. Frank (Ed.), Springer-Verlag, Chapter 12, pp. 391-449, 1999.

[2] Cherrualt, Y. Optimisation: méthodes locales et globales, Presses Universitaires de France (PUF), Collections mathématiques, France, 1999.

[3] González E., Carvajal K, Acuña G, (2002), Modelamiento del Proceso de Molienda Semiautógena a Través de Redes Neuronales, Actas en CD del *Encuentro Chileno de Computación,* Copiapó, Chile, Noviembre 2002.

[4] Hornik, K., Stinchcombe, M. Y White, H., (1989), Multilayer feddforward networks are universal approximators, *Neural Networks*, **2**:359-366

[5] Jamet Marcela, San Juan Enrique, "Identificación de un Sistema Multivariable que modela el proceso de molienda SAG", Trabajo de Doctorado en ciencias de la ingeniería mención automática, Universidad de Santiago de Chile.

[6] Lennox, B., Montague, G., Frith, A., Gent, C., Bevan, V. Industrial application of neural networks – an investigation, *Journal of Process Control* 11, pp. 497-507, 2001.

[7] Magne Luis, "Estudio del transporte de masa en molinos semiautógenos", Tesis de Doctorado en Ciencias de la Ingeniería con mención en Metalurgia, Universidad de Concepción, 1999.

[8] Magne L., Améstica R.,Barría J. y Menacho J., (1995). Dynamic modelling of semiautogenous milling based on a simplified phenomenological model, *Rev. Metal. Madrid*, **31**(2):97-105 (in Spanish).

[9] Magne, L., Valderrama, W., "Operación y mantención en plantas de molienda semiautógena", Tesis de Doctorado, Universidad de Santiago de Chile, 1997.

[10] Soroush, M. (1998), State and parameter estimations and their applications in process control, *Comput. Chem. Engng.*, **23**:229-245.

[11] Valdés-González, H., Flaus, J, Acuña G., Moving horizon state estimation with global convergence using interval techniques: application to biotechnological processes, Journal of Process Control, 13(4), pp. 325-336, 2003

Hybrid Learning Systems Based on Support Vector Machines and Radial Basis Function Neural Networks

Haydemar Núñez
Laboratorio de Inteligencia Artificial
Facultad de Ciencias. Universidad Central de Venezuela. Caracas, Venezuela
hnunez@strix.ciens.ucv.ve

Cecilio Angulo, Andreu Català,
ERIC – Engineering & Research in Computational Intelligence
Technical University of Catalonia**.** Vilanova i la Geltrú, Spain
cecilio.angulo@upc.es, andreu.catala@upc.es,

Abstract

Two methods are proposed for the symbolic interpretation of both Support Vector Machines (SVM) and Radial Basis Function Neural Networks (RBFNN). These schemes, based on the combination of support vectors and prototype vectors by means of geometry, produce rules in the form of ellipsoids and hyper-rectangles. Results obtained from a certain number of experiments on artificial and real databases in different domains allow conclusions to be drawn on the suitability of our proposal. Moreover, schemes that incorporate the available prior domain knowledge expressed as symbolic rules into SVMs are explored, with excellent performances being obtained.
Keywords: Artificial Intelligence, Support Vector Machines, Neural Networks, Hybrid Architectures

Resumen

En este trabajo se proponen dos métodos para la interpretación simbólica de máquinas de soporte vectorial (SVM) y redes neuronales de función de base radial (RBFNN), respectivamente. Ambos esquemas se basan en la combinación, mediante geometría, de los vectores de soporte generados por una SVM y vectores prototipos o centros de una RBFNN, para producir descripciones en la forma de elipsoides e hiper-rectángulos. Los resultados de los numerosos experimentos realizados sobre bases de datos artificiales y reales de diferentes dominios, nos permiten concluir sobre la viabilidad de la propuesta. También, se exploran esquemas para la inserción, en máquinas de soporte vectorial, del conocimiento previo disponible expresado como reglas simbólicas.
Palabras claves: Inteligencia Artificial, Máquinas de Soporte Vectorial, Redes Neuronales, Arquitecturas Híbridas

1. INTRODUCTION

When neural networks are used for constructing classifiers, black-box type models are generated, even if the obtained performance is good. With the aim of facilitating the interpretation of the classifier system, over the last 10 years rule extraction methods for trained neural networks have been developed [1],[6],[14],[21] and [22]. In the case of radial basis function neural networks (RBFNN) [11],[15],[19], the rule extraction algorithms proposed usually impose restrictions over training in order to avoid overlapping between classes or categories and thus facilitate the extraction process [4],[9],[10] and [13].

On the other hand, over the last 7 years it has been demonstrated that the support vector machines (SVM) [5],[7],[11] derived from V.N. Vapnik's Statistical Learning Theory [23], have excellent classification and approximation qualities for all kinds of problems. However, as in the case of the neural networks, the models generated by these machines are difficult to understand from a user's point of view.

In order to interpret these models, the present work studies the classifier function structure of the SVM and the RBFNN in depth [17]. Starting with the support vectors selected by a SVM and prototype vectors generated by any clustering algorithm or by RBFNN, a rule extraction method for SVM is proposed. This method produces descriptions in the form of ellipsoids and it is independent of the type of training used. Furthermore, the interpretation of the rules is facilitated through a second derivation in the form of hyper-rectangles. As an original contribution and extension of the work, the algorithm is modified for the exclusive use of the information supplied by a SVM, thus achieving the elimination of the intrinsic variability in the classification made by clustering algorithms or by RBFNN.

Additionally, starting with the RBFNN centres and using support vectors as class delimiters, a rule extraction method for RBFNN is proposed. As in the previous case, it produces descriptions in the form of ellipsoids and hyper-rectangles. This method solves the overlapping between classes without imposing restrictions on the network architecture or its training regime.

Finally, since the final information will be expressed in the form of interpretable rules, a third contribution of this work is the analysis of schemes for inserting knowledge into SVMs, when this knowledge is available in form of rules. The alternatives studied allow us to conclude that access to this information by the SVM improves its final performance.

This paper is organized as follows: the rule extraction method from trained SVMs is described in the next section, along with the experimental results. Section 3 presents the rule extraction method for RBFNN. Section 4 describes prior knowledge integration methods into SVM. Finally, we present the conclusions and future work.

2. INTERPRETATION OF SUPPORT VECTOR MACHINES

The solution provided by a SVM is a summation of kernel functions constructed on the base of the support vectors

$$f_a(\boldsymbol{x}) = sign\left(\sum_{i=1}^{sv} \alpha_i y_i K(\boldsymbol{x}, \boldsymbol{x}_i) + b \right) \tag{1}$$

The support vectors (***sv***) are the data nearest the separation limit between classes. They are the most informative samples for the classification task. By using geometric methods, these vectors with prototype vectors (generated by any clustering algorithm or by RBFNN) are combined in order to construct regions in the input space, which are later translated to if-then rules. These regions can be of two types: ellipsoids and hyper-rectangles. Ellipsoids generate rules whose antecedent is the mathematical equation of the ellipsoid. Hyper-rectangles (defined from parallel ellipsoids to the axes) generate rules whose premise is a set of restrictions on the values of each variable (Figure 1).

The ellipsoids must adjust to the form of the decision limit. In order to obtain one ellipsoid with these characteristics, the support vectors are used to determine the axes and vertices as follows: first, the algorithm determines one prototype vector per class by using a clustering algorithm. This vector will be the centre of the ellipsoid. Then, a support vector of the same class with a value α smaller than parameter C and with the maximum distance to the prototype is chosen. The straight line defined by these two points is the first axis of the ellipsoid. The rest of the axes and the associated vertices are determined by simple geometry. To construct hyper-rectangles, a similar procedure is followed. The only difference is that lines parallel to the axes are used to define the axes of the associated ellipsoid.

The number of regions necessary to describe the SVM model will depend on the form of the decision limit. One ellipsoid may not be sufficient to describe the data. Then, the rule base is determined by an iterative procedure that begins with the construction of a general ellipsoid, which is divided into ellipsoids that adjust progressively to the form of the surface decision determined by SVM. To determine when to divide an ellipsoid, a partition test is applied. If the test result is positive for one ellipsoid, the latter is divided. We consider a partition test to be positive if the generated prototype belongs to another class, if one of the vertices belongs to another class or if a support

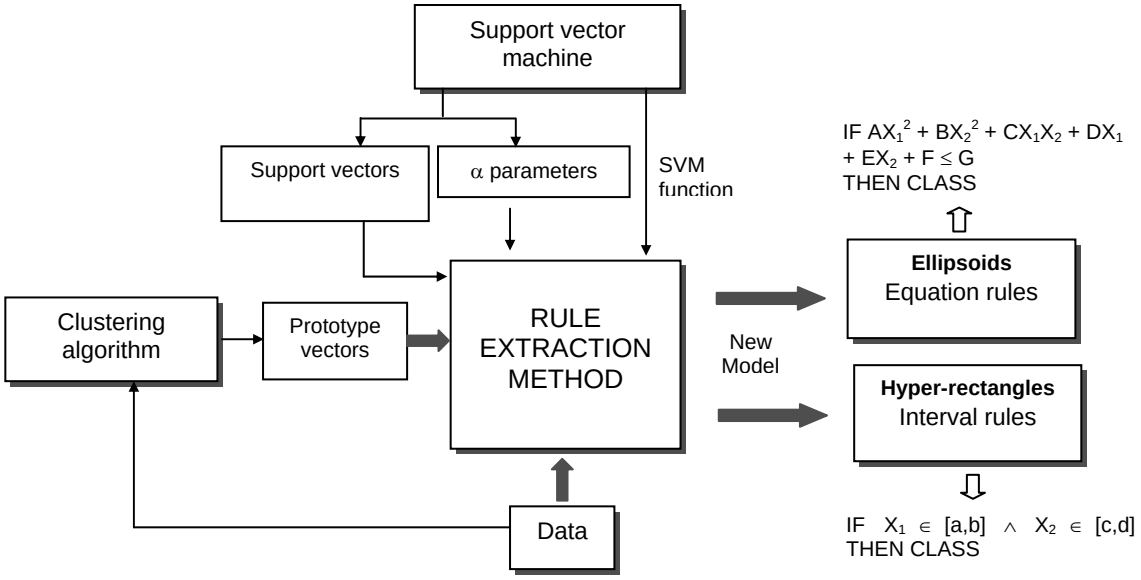


Figure 1. Rule extraction method for SVMs

vector from another class exits within the region. To determine the class label of the prototypes and the class label of the vertices, the SVM function is used.

Then, in order to define the number of rules per class the algorithm proceeds as follows:

Beginning with a single prototype, the associated ellipsoid is generated. Next, the partition test is applied on this region. If it is negative, the region is translated to a rule. Otherwise, new regions are generated. In this form, each iteration produces *m* regions with a positive partition test and *p* regions with a negative partition test. The latter are translated into rules. In the next iteration, the data of the *m* regions are used to determine *m+1* new prototypes and to generate *m+1* new ellipsoids. This procedure is stopped once all partition tests are negative or the maximum number of iterations has been reached. This process allows the number of rules generated to be controlled.

After the rules are extracted, the system classifies an example by assigning it to the class of the nearest rule in the knowledge base (following the nearest-neighbour philosophy) by using the Euclidian distance. If an example is covered by several rules, we choose the class of the most specific ellipsoid or hyper-rectangle containing the example; that is, the one with the smallest volume. Figure 2 shows an example of regions generated for each iteration.

2.1 Experiments

In order to evaluate the performance of the rule extraction algorithm, we carried out two kinds of experiments: with artificial data sets and databases obtained from the UCI repository [3]. The algorithms associated to the extraction method were developed on Matlab v5.3. The training of the SVMs on the artificial data sets was carried out with "Matlab SVM/K-SVCR Toolbox" software [2]; for databases from the repository UCI we used "OSU Support Vector Machines Toolbox v3.00" software [12]. We used the k-means clustering algorithm [8] to generate the prototype vectors.

Because the space is limited, only the experiments on UCI databases are described. Table 1 shows the characteristics of the databases that were used. The performance of the rules generated was quantified using the following measures:

- *Error (Err)*: This is the classification error provided by the rules on the test set.
- *Consistency (Cn)*: This is the percentage of the test set for which the network and the rule base output agree.
- *Coverage (Cv)*: This is the percentage of examples from the test set covered by the rule base.
- *Overlapping (Ov)*: This is the percentage of examples from the test set covered by several rules.
- *Number of extracted rules (NR).*

Table 2 shows the prediction error of the SVM and the performance values of the extracted rule base for each problem. The results were obtained by averaging over stratified ten-fold cross-validation. It should be emphasized that the consistency percentage between the rule base and the SVM is very high. These values indicate that the rule base captures most of the information embedded in the SVM.

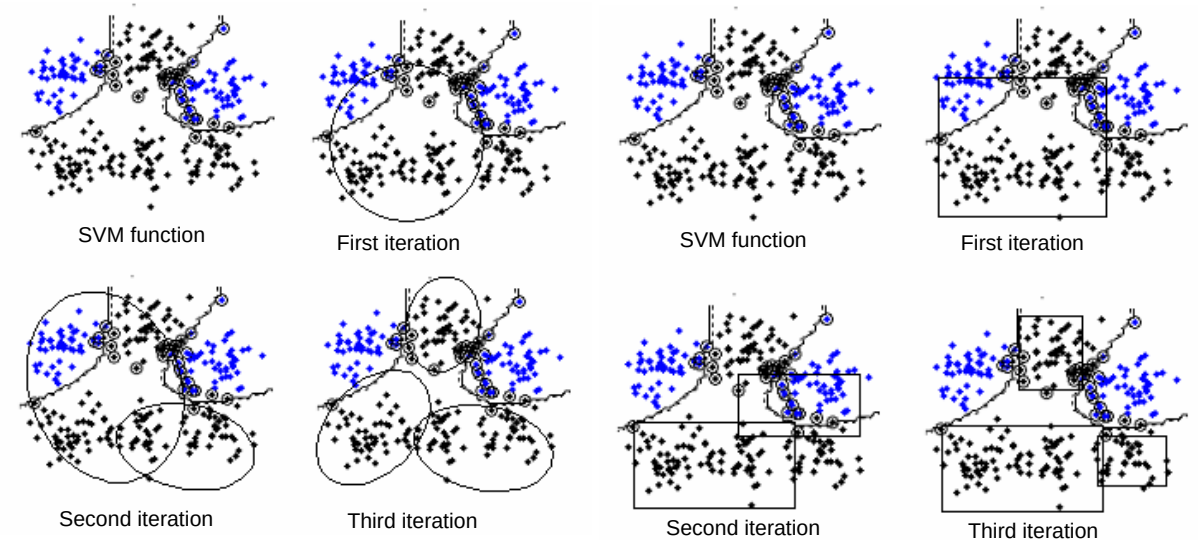


Figure 2. Regions generated by the rule extraction method
(a) Ellipsoids. (b) Hyper-rectangles

Table 1. Databases and their characteristics

ID	Database	Data	Attributes	Type of attributes	Classes
1	IRIS	150	4	Real	3
2	WISCONSIN	699	9	Symbolic	2
3	WINE	178	13	Real	3
4	SOYBEAN	47	35	Integer	4
5	New-THYROID	215	5	Real	3
6	AUSTRALIAN	690	14	Real and symbolic	2
7	SPECT	267	23	Binary	2
8	MONK1	432	6	Symbolic	2
9	MONK2	432	6	Symbolic	2
10	MONK3	432	6	Symbolic	2
11	ZOO	101	16	Symbolic	7
12	HEART	270	13	Real, symbolic and binary	2

On the other hand, it was observed that the quality of the solution depends on the initial values for the centres; the selection of prototypes affects the number and quality of the extracted rules. Therefore, it was necessary to apply k-means several times, starting with different initial conditions and then choosing the best solution. Although the dependency of the clustering final result on the random way the prototypes are selected is well known, this characteristic is never desirable. The solution to this problem is an opened working area and a new proposal such as the following one presented here means positive alternatives.

2.2 Overcoming the Randomness of Clustering Algorithms

In order to eliminate the sensitivity of the rule extraction method to the initial conditions of clustering, we proposed the initial centres for the clustering algorithms from the support vectors should be determined. Thus, if *m* is the number of necessary prototypes for iteration, then the *m* initial conditions for k-means are determined in the following form:

By each class

- Select *m* support vectors with same class label.
- Assign examples to their closest support vector according to the Euclidean distance function.

Table 2. Performance values obtained for each database.

ID database	Error SVM	Equation rules					Interval rules				
		Error	Cn.	Cv.	Ov.	NR	Error	Cn.	Cv.	Ovl.	NR
1	0.033	0.040	98.00	72.00	0.67	7.0	0.040	99.33	68.00	0.00	4.7
2	0.031	0.034	98.52	89.15	0.30	4.0	0.037	98.24	93.26	0.73	5.1
3	0.022	0.017	98.30	67.49	0.55	5.9	0.023	96.07	69.89	2.28	8.2
4	0.000	0.000	100.00	33.00	0.00	6.3	0.020	98.00	75.00	0.00	6.4
5	0.032	0.032	97.21	80.07	0.00	7.1	0.032	96.30	70.58	2.72	9.2
6	0.127	0.133	93.60	65.70	2.86	18.4	0.137	93.20	87.66	3.18	21.6
7	0.102	0.117	96.26	21.39	0.53	14.0	0.112	96.26	40.11	0.00	22.0
8	0.051	0.091	85.18	33.56	0.00	24.0	0.056	92.59	59.49	0.00	33.0
9	0.178	0.211	76.38	32.87	0.46	60.0	0.219	75.95	63.19	5.78	84.0
10	0.023	0.034	97.45	27.55	0.00	7.0	0.027	99.07	100.00	0.00	4.0
11	0.045	0.045	99.09	31.45	0.74	9.8	0.052	97.07	74.97	0.00	9.4
12	0.159	0.137	97.04	56.67	0.74	4.5	0.163	96.67	60.01	0.00	20.4

- Once the initial partitions have been established, the mean of all instances in each partition is calculated.
- These points are the initial conditions for the clustering algorithm.

Three criteria were used to select the support vectors:

- *Partition scheme 1*: select those vectors with the smallest average dissimilarity with respect to the data.
- *Partition scheme 2*: select the support vectors nearest to each other.
- *Partition scheme 3*: organize the support vectors in descending order according to α parameter and select the *m* first vectors.

In all cases, if more prototypes are required than there are support vectors available, the assembly of initial partitions is completed using those points with the smallest average dissimilarity with respect to the data.

These schemes were evaluated on trained SVMs on the same databases from the UCI repository. Tables 3, 4 and 5 show the obtained results when these partition schemes were used. We can observe that the results are comparable with those obtained by using k-means. Thus, it is possible to obtain a good rule base with a single application of the rule extraction algorithm.

Table 3. Performance values for each database using partition scheme 1

ID database	Error SVM	Equation rules					Interval rules				
		Error	Cn.	Cv.	Ov.	NR	Error	Con.	Cub.	Sol.	NR
1	0.033	0.040	99.33	67.33	1.33	7.0	0.047	98.67	67.33	1.33	4.7
2	0.031	0.032	98.07	87.83	0.14	4.4	0.032	98.53	89.74	1.02	10.0
3	0.022	0.023	96.63	67.91	1.11	6.0	0.028	97.18	75.34	3.89	10.9
4	0.000	0.020	98.00	25.00	0.00	4.9	0.020	98.00	70.00	2.00	7.7
5	0.032	0.028	96.73	76.32	2.74	8.0	0.042	96.30	72.03	3.18	10.2
6	0.127	0.131	91.00	59.98	2.75	21.1	0.146	92.75	84.91	4.47	24.0
7	0.102	0.182	88.77	19.79	5.88	12.0	0.118	90.37	71.23	1.60	34.0
8	0.051	0.123	86.34	52.91	0.23	28.0	0.141	86.34	69.91	6.48	34.0
9	0.178	0.201	78.24	34.03	1.38	64.0	0.231	78.00	56.71	2.31	72.0
10	0.023	0.025	96.99	40.05	0.00	8.0	0.018	99.53	95.60	0.00	8.0
11	0.045	0.046	99.09	29.23	0.00	10.2	0.052	95.96	74.97	0.00	9.5
12	0.159	0.151	96.30	52.22	0.37	4.6	0.167	96.30	57.04	0.37	21.0

Table 4. Performance values for each database using partition scheme 2

ID database	Error SVM	Equation rules					Interval rules				
		Error	Cn.	Cv.	Ov.	NR	Error	Con.	Cub.	Sol.	NR
1	0.033	0.047	98.67	69.33	0.67	6.7	0.040	99.33	68.67	0.67	4.8
2	0.031	0.035	98.38	89.15	0.29	4.3	0.035	98.09	91.36	0.73	9.3
3	0.022	0.028	98.06	63.36	0.55	7.2	0.028	97.22	73.64	1.11	11.7
4	0.000	0.020	98.00	25.00	0.00	4.9	0.020	98.00	68.00	2.00	8.2
5	0.032	0.032	96.28	80.52	0.93	7.9	0.042	95.37	69.16	3.65	10.6
6	0.127	0.149	90.85	65.16	3.90	23.2	0.157	90.01	88.39	7.63	25.9
7	0.102	0.176	90.37	17.65	0.00	20.0	0.197	89.30	36.89	3.21	22.0
8	0.051	0.171	84.25	59.03	3.24	12.0	0.078	90.74	90.51	13.4	15.0
9	0.178	0.215	74.53	33.10	1.62	70.0	0.231	78.00	56.71	2.31	72.0
10	0.023	0.048	96.99	36.00	0.00	9.0	0.018	99.54	94.90	0.00	9.0
11	0.045	0.046	89.09	27.11	0.00	10.1	0.052	95.96	74.97	0.00	9.5
12	0.159	0.137	95.56	54.44	0.74	4.8	0.166	95.18	57.78	0.74	20.3

Table 5. Performance values for each database using partition scheme 3

ID database	Error SVM	Equation rules					Interval rules				
		Error	Cn.	Cv.	Ov.	NR	Error	Con.	Cub.	Sol.	NR
1	0.033	0.013	96.67	62.00	0.00	4.0	0.033	96.67	72.00	0.00	5.0
2	0.031	0.036	97.35	87.83	0.00	4.5	0.035	98.39	91.80	0.59	9.0
3	0.022	0.023	97.74	64.62	0.56	7.0	0.025	97.75	69.18	0.56	14.6
4	0.000	0.00	100.00	15.50	0.00	5.8	0.00	100.00	70.50	6.50	6.8
5	0.032	0.032	97.21	78.27	0.95	8.6	0.033	96.30	71.69	0.93	11.2
6	0.127	0.142	92.74	63.42	2.89	21.3	0.142	90.43	81.31	3.91	34.5
7	0.102	0.123	95.72	17.11	0.53	16.0	0.117	92.51	33.12	2.14	28.0
8	0.051	0.129	86.11	37.13	0.23	12.0	0.044	92.82	88.99	6.48	15.0
9	0.178	0.213	78.34	31.48	0.00	61.0	0.196	78.70	62.26	2.31	65.0
10	0.023	0.056	94.67	44.90	0.00	5.0	0.023	99.07	90.74	0.00	10.0
11	0.045	0.053	98.32	38.88	0.00	10.2	0.059	96.10	73.09	0.00	9.3
12	0.159	0.167	97.78	51.11	0.00	5.2	0.167	95.56	60.00	0.00	21.4

3. INTERPRETATION OF RADIAL BASIS FUNCTION NEURAL NETWORKS

The hypothesis space implanted by these learning machines is constituted by functions of the form

$$f(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{v}) = \sum_{k=1}^{m} w_k \phi_k (\boldsymbol{x}, \boldsymbol{v}_k) + w_0 \tag{2}$$

The nonlinear activation function ϕ_k expresses the similarity between any input pattern $\mathbf{x}$ and the centre $\mathbf{v}_k$ by means of a distance measure. Each function ϕ_k defines a region in the input space (receptive field) on which the neuron produces an appreciable activation value. In the common case when the Gaussian function is used, the center $\mathbf{v}_k$ of the function ϕ_k defines the prototype of input cluster *k* and the variance σ_k the size of the covered region in the input space.

The local nature of RBF networks makes them an interesting platform for performing rule extraction. However, the basis functions overlap to some degree in order to give a relatively smooth representation of the distribution of training data [11],[15]. This overlapping is a shortcoming for rule extraction.

Few rule extraction methods for RBFNN have been developed [4],[9],[10] and [13]. In order avoid the overlapping, most of them use special training regimes or special architectures so as to guarantee that the RBF nodes are assigned and used by a single class. Our proposal for rule extraction does not suppose any training methods or special architecture; it extracts rules from an ordinary RBFNN. In order to solve the overlapping, a support vector machine (SVM) is used as a frontier pattern selector.

The rule extraction method for RBFNN derives descriptions in the form of ellipsoids and hyper-rectangles. Therefore, the algorithm for constructing an ellipsoid for SVM is used; the RBF centres replace the prototypes. In this case, support vectors establish the boundaries between classes.

Initially, by assigning each input pattern to its closest centre of RBF node according to the Euclidean distance function, a partition of the input space is made. When assigning a pattern to its closest centre, the former will be assigned to the RBF node that will give the maximum activation value for that pattern. From these partitions the ellipsoids are constructed.

Next, a class label is assigned for each centre of RBF units. The output value of the RBF network for each centre is used in order to determine this class label.

Then, an ellipsoid with the associated partition data is constructed for each node. Once the ellipsoids have been determined, they are transferred to rules.

This procedure will generate a rule for each node. Nevertheless, other classes of data could be present in the partition of the RBF unit. For these data we determine the mean of each class. Each mean is used as a centre of its class in order to construct an ellipsoid with the associated data.

In order to eliminate or to reduce the overlapping that could exist between ellipsoids of different classes, an overlapping test is applied. Overlapping tests verify whether a support vector from another class exits within the ellipsoid. Because the support vectors are the points nearest to the decision limit, the presence of these vectors within an ellipsoid of a different class is a good indicator of overlapping. If the overlapping test is positive, the ellipsoid is divided.

This procedure will allow the rule base to be refined in order to reduce the overlapping between classes. When the ellipsoids are divided, more specific rules are generated to exclude data from other classes. This procedure can be executed in an iterative form; depending of the number of iterations, two or more partitions by ellipsoids can be obtained. The user can establish the maximum number of iterations. Thus, it is possible to control the number of rules generated by the RBF node.

3.1 Experiments

In order to evaluate the performance of the rule extraction algorithm, we carried out two kinds of experiments with artificial data sets and databases obtained from the UCI repository. The algorithms associated to the extraction method were developed on a Matlab v5.3. Again, only experiments on databases from the UCI repository are described.

We used 6 databases from this repository and the same performance parameters. With the purpose of validating the hypothesis of the rule extraction method independent of the training techniques used, two different training procedures were used: the Netlab software [16], which uses the EM algorithm to determine the RBF centres, and the Orr software [18], which uses forward selection.

Tables 6 and 7, show the prediction error of the RBF network and the performance values of the extracted rule base. Results were obtained by averaging over stratified ten-fold cross-validation. We can observe a high agreement between the results obtained from the rule base and those obtained from the RBF network. However, because the Orr method needs to use more hidden units to obtain a better performance, it produces a greater rule base.

Table 6. Results obtained from data sets (with Netlab software)

ID	RBF nodes	RBF error	Equation rules					Interval rules				
			Err	Cn.	Cv.	Ov.	NR	Err	Cn.	Cv	Ov	NR
1	4.5	0.040	0.033	96.67	64.67	0.00	6.8	0.040	97.33	70.67	0.00	6.5
2	2.2	0.029	0.032	98.24	91.21	1.76	5.7	0.041	96.78	95.01	2.93	14.5
3	3.0	0.011	0.038	97.78	66.43	2.75	9.7	0.046	95.38	81.30	7.32	10.7
4	5.4	0.020	0.020	96.00	17.00	0.00	6.9	0.060	91.50	62.50	4.00	10.2
5	9.3	0.065	0.060	94.91	80.99	2.29	16.3	0.056	94.44	79.07	6.06	16.5
10	6.0	0.048	0.060	95.14	63.19	0.93	14.0	0.027	97.91	100.0	0.00	11.0

Table 7. Results obtained from data sets (with Orr software)

ID	RBF nodes	RBF error	Equation rules					Interval rules				
			Err	Cn.	Cv	Ov.	NR	Err	Cn.	Cv.	Ov.	NR
1	5.1	0.033	0.027	96.67	72.00	0.00	9.2	0.033	94.67	74.67	0.00	8.9
2	21.5	0.034	0.035	97.22	83.41	0.43	26.4	0.049	96.19	95.31	3.95	28.2
3	15.2	0.039	0.039	93.26	63.20	0.62	38.1	0.062	93.30	85.38	7.26	86.2
4	12.4	0.000	0.040	96.00	30.00	0.00	14.7	0.085	91.50	91.00	30.50	19.6
5	29.2	0.062	0.064	88.87	61.41	0.45	33.0	0.054	88.87	72.23	0.90	33.0
10	12.0	0.050	0.069	90.97	63.19	3.93	23.0	0.064	92.36	100.0	57.40	33.0

4. INSERTING PRIOR KNOWLEDGE IN SUPPORT VECTOR MACHINES

To insert prior knowledge into the support vector machines, different techniques have been used: by means of generated virtual examples from transformation functions applied to the data [26] or the support vectors [20], designing kernel functions that adapt to the problems [7], [24], and adding new restrictions of the optimization problem [25]. Nevertheless, sometimes the prior knowledge is difficult to formalize as a transformation or kernel function; this knowledge can be expressed as a set of symbolic rules, which experts give as follows:

$$IF\ \ x_1 \in [a_1, b_1] \wedge x_2 \in [a_2, b_2] \wedge \ldots \wedge x_m \in [a_m, b_m]\ \ THEN\ \ class$$

How could we integrate this knowledge in a SVM in this case? We propose doing it by means of a strategy similar to the virtual example method.

The prior knowledge expressed as a rule defines a convex region in the input space (in the form of a hyper-rectangle). This convex region is defined by a set of vertices, which provide information on the limits of the associated rule. These vertices can be used as virtual examples and added to the learning set. However, these samples would be treated specially because the knowledge that they provide is correct. Then, given the training set $D = \{(\boldsymbol{x}_i, y_i) i = 1..n\}$ and the virtual example set $V = \{(\boldsymbol{x}_j, y_j) j = 1..d, d = 2^m\}$, three schemes are proposed for incorporating this knowledge into a SVM.

- *Insertion method 1*: A new restriction is added to the optimization problem, which supposes that the virtual examples can be classified without error:

$$\begin{aligned}
\text{Minimize} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}\xi_i \\
\text{Subject to} \quad & y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i + b) \geq 1 - \xi_i \qquad (\boldsymbol{x}_i, y_i) \in D, i = 1..n \\
& y_j(\boldsymbol{w}\cdot\boldsymbol{x}_j + b) \geq 1 \qquad (\boldsymbol{x}_j, y_j) \in V, j = 1..d \qquad \leftarrow \textbf{New restriction} \\
& \xi_i \geq 0 \quad \forall i
\end{aligned}$$

- *Insertion method 2*: Two different parameters for error control are defined

$$\begin{aligned}
\text{Minimize} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C_d\sum_{i=1}^{n}\xi_i^d + C_v\sum_{j=1}^{d}\xi_j^v \\
\text{Subject to} \quad & y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i + b) \geq 1 - \xi_i^d \qquad (\boldsymbol{x}_i, y_i) \in D, i = 1..n \\
& y_j(\boldsymbol{w}\cdot\boldsymbol{x}_j + b) \geq 1 - \xi_j^v \qquad (\boldsymbol{x}_j, y_j) \in V, j = 1..d \qquad \leftarrow \textbf{New restriction} \\
& \xi_i^d \geq 0 \quad \forall i \\
& \xi_j^v \geq 0 \quad \forall i
\end{aligned}$$

The C_d y C_v parameters determine a balance between the information from the training data and the prior knowledge.

- *Insertion method 3:* Train a SVM with the vertices of the rules of one class and the data from other classes. This pre-processing generates a set of virtual supports by class. Next, train a SVM with all the learning sets and the generated virtual support vectors from this pre-processing. In this form, the part of the prior knowledge that would be more informative for the classification task is added to the data (the vertices nearest to the surface limit).

4.1 Experiments

In order to evaluate the rule insertion methods, we applied them to the three MONK problems from the UCI repository, because they have a defined domain theory. To verify whether it is possible to improve the SVM performance when inserting the domain knowledge, the following procedure was carried out: first, a SVM without added knowledge was trained. We then trained a SVM using the rule insertion methods. This procedure was repeated 50 times and we determined the average values on the following parameters:

- Training set error (ErrEnt)
- Test set error (ErrTest)
- Number of support vectors from MONK class (Sv1)
- Number of support vectors from NO-MONK class (Sv2)

The used values of the C_d and C_v parameters guarantee a greater weight to the errors associated to the virtual examples. Table 8 shows the results obtained. We can observe that the rule insertion methods improve the original SVM performance.

Table 8. Results obtained by applying the insertion methods on MONK databases.

Strategy	MONK1				MONK2				MONK3			
	Err Ent	Err Test	SV1	SV2	Err Ent	Err Test	SV1	SV2	Err Ent	Err Test	SV1	SV2
Within knowledge	0	0.059	28.24	23.22	0.008	0.146	34.22	29.74	0.023	0.054	14.04	14.44
Method 1	0	0.032	40.08	23.80	0.003	0.088	47.52	30.80	0.021	0.034	14.00	13.36
Method 2	0	0.032	40.08	23.80	0.003	0.088	47.84	30.72	0.020	0.033	13.04	13.70
Method 3	0	0.029	38.44	23.96	0.003	0.086	44.20	30.90	0.021	0.029	13.52	13.28

5. CONCLUSIONS AND FUTURE WORK

With the aim of providing SVMs with explanation power, a method that converts the knowledge embedded in a trained SVM into a representation based on rules was developed. The experiments with the rule extraction method on artificial data sets and with databases of different domains, show high levels of equivalence between the SVM and the extracted rule base.

Additionally, a rule extraction method for RBFNN was developed, which uses the algorithm for constructing an ellipsoid proposed for SVM as a core. Based on the results obtained, it can be concluded that the extraction technique derives consistent models with the RBFNN without any previous requirement from either the used training regime or its architecture.

The possibility of adding prior knowledge expressed as symbolic rules in a SVM was established using schemes based on virtual examples, which are generated from the associated vertices with hyper-rectangles related with the rules.

Given the achievements of this work, it is possible to raise new problems. For example, it would be interesting to study ways of extending the rule extraction methods to regression problems. If this were achieved, a more versatile technique would be available for a larger number of cases. Another question currently emerging is the study of the possibility of using another representation language to express the new model, such as fuzzy rules generated by ellipsoids.

References

[1] Andrews R., Diederich J. and Tickle A. (1995). A survey and Critique of Techniques for Extracting Rules from Trained Artificial Neural Networks. *Knowledge-Based Systems*. 8(6):373-389.

[2] Angulo, C. (2001). Matlab SVM/K-SVCR Toolbox. http://webesaii.upc.es/usr/cecilio/software.htm.

[3] Blake, C.L. and Merz, C.J. (1998). UCI Repository of Machine Learning Data-Bases. University of California, Irvine. http://www.ics.uci.edu/~mlearn/MLRepository.html.

[4] Brotherton, T., Chadderdon, G. and Grabill, P. (1999). Automated Rule Extraction for Engine Vibration Analysis. *Proc. IEEE Aerospace Conference*. 3:29-38.

[5] Cortes C. and Vapnik V. (1995). Support-Vector Networks. *Machine Learning*. 20:273-297.

[6] Craven M. and Shavlik J. (1997). Using Neural Networks for Data Mining. *Future Generation Computer Systems*. 13:211-229.

[7] Cristianini, N. and Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press.

[8] Duda, R., Hart, P. and Stork, D. (2001). *Pattern Recognition*. Second Edition. John Wiley & Sons, Inc.

[9] Fu, X. and Wang, L. (2001). Rule Extraction by Genetic Algorithms Based on a simplified RBF Neural Network. *Proc. of the Congress on Evolutionary Computation*. 2:753-758.

[10] Huber, K. and Berthold, M. (1995). Building Precise Classifiers with Automatic Rule Extraction. *Proc. IEEE International Conference on Neural Networks*. 3:1263-1268

[11] Kecman, V. (2001). *Learning and Soft Computing. Support Vector Machines, Neural Networks and Fuzzy Logic Models*. MIT Press.

[12] Ma, J. and Zhao, Y. (2002). OSU Support Vector Machines Toolbox, version 3.0. http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[13] McGarry, K., Wermter, S. and MacIntyre, J. (2001). Knowledge Extraction from Local Function Networks. *Proc. International Joint Conference on Neural Networks*. 765-770

[14] Mitra, S., Pal, S. K. and Mitra, P. (2002). Data Mining in Soft Computing Framework: A survey. *IEEE Transactions on Neural Networks*. 13 (1):3-14.

[15] Moody, J. and Darken, C.J. (1989). Fast Learning in Networks of Locally-Tuned Processing Units. *Neural Computation*. 1:281-294.

[16] Nabney, I. and Bishop, C. Netlab Neural Networks Software. http://www.ncrg.aston.ac.uk/netlab.

[17] Núñez, H. (2003). Hybrid Learning Systems based on Support Vector Machines and Radial Basis Function Neural Networks. Ph.D dissertation. Technical University of Catalonia. Spain.

[18] Orr, M. Radial Basis Function Networks. http://www.anc.ed.ac.uk/~mjo/rbf.html.

[19] Poggio, T. and Girosi, F. (1990). Networks for Approximation and Learning. *Proceedings of the IEEE*. 78:1481-1497.

[20] Schölkopf, B., Burges, C. and Vapnik, V. (1996). Incorporating Invariances in Support Vector Learning Machine. *Lecture Notes in Computer Science*. 1112:47-52.

[21] Setiono, R., Leow, W. and Zurada, J. (2002). Extraction Rules from Artificial Neural Networks for Nonlinear Regression. *IEEE Transactions on Neural Networks*. 13(3):564-577.

[22] Tickle, A., Maire, F., Bologna, G., Andrews, R.; Diederich, J.: Lessons from Past, Current Issues, and Future Research Directions in Extracting the Knowledge Embedded Artificial Neural Networks. In: Wermter, S. and Sun, R. (Eds): *Hybrid Neural System*s. Springer-Verlag.

[23] Vapnik, V. (1998). *Statistical Learning Theory*. John Wiley & Sons, Inc.

[24] Zien, A., Rätsch, G., Mika, S., Schölkopf, B., Lengauer, T. and Müller, K.-R. (2000). Engineering Support Vector Machine kernels that Recognize Translation Initiation Sites. *Bioinformatics*. 16(9):799-807.

[25] Zhang, X. (1999). Using Class-Center vectors to build Support Vector Machines. *Proc. IEEE Conference on Neural Networks for Signal Processing*. 3-11.

[26] Zhao, Q. and Principe, J.C. (1999). Improving ATR Performance by Incorporating Virtual negative examples. *Proc. International Joint Conference on Neural Networks*. 5:3198-3203.

Un soporte de comunicación grupal para agentes móviles

Guillermo Rigotti
UNICEN – Fac. de Ciencias Exactas-ISISTAN
Pje. Arroyo Seco, (7000) Tandil, Bs. As. Argentina
e-mail: grigotti@exa.unicen.edu.ar

Resumen

En los últimos años se ha producido un desarrollo significativo en el área de agentes móviles. Estos sistemas han demostrado su aptitud para un conjunto diverso de aplicaciones de gran uso en la Internet.
Si bien se han abordado con éxito diferentes aspectos de dichos sistemas, logrando soluciones concretas, se ha dejado de lado el aspecto de la comunicación, referido al transporte de la información a intercambiar entre los agentes móviles. Los mecanismos utilizados se han heredado principalmente de la teoria de sistemas distribuidos, no adaptándose completamente a las características de la comunicación entre agentes móviles. Es así que gran parte de los sistemas en uso cubren redes locales o intranets, o bien soportan sólo una cantidad limitada de agentes. Un problema aún abierto a la investigación lo constituye el hecho de encontrar un soporte de comunicación y localización adaptable a sistemas de agentes móviles que involucran gran cantidad de agentes dispersos en áreas extensas de la Internet. Se han realizado intentos de implementar este soporte utilizando transmisión multicas;, sin embargo, cuando éste es implementado a nivel de red, presenta limitaciones en cuanto a su despliegue en la totalidad de la red y en cuanto al servicio ofrecido, en particular a los agentes móviles.
En este trabajo se propone una infraestructura de comunicación para agentes móviles, que está basada en un soporte de transmisión multicast, implementado en los servidores de agentes, que permite satisfacer las demandas de las aplicaciones a un costo razonable y con una performance aceptable
Palabras clave: Agentes móviles, Multicast.

Abstract

In the last years a significant development has taken place in the area of mobile agents. These systems have demonstrated their aptitude for a diverse group of applications of great use in the Internet. Although different aspects of this systems have been approached with success, achieving concrete solutions, it has been left aside the aspect of the communication, related to the transport of the information to be exchanged among the mobile agents. The used mechanisms have been inherited mainly of the theory of distributed systems, not adapting completely to the communications requirements posed by mobile agents. Due to this problem, this kind of applications are either restricted to be deployed in intranets or to support only a limited number of agents. A problem even open to the investigation is to develop a support for communication among and localization of mobile agents, that involves a great number of agents moving in vast areas of the Internet. Some approaches have been carried out aimed to implement this kind of support using the existent infrastructure for multicast transmission. However, because it is implemented at network level, it presents limitations in several aspects, mainly its deployment all over the Internet and the service offered to the mobile agents. In this work we propose a communication infrastructure for mobile agents that is based on a multicast support implemented in the agent servers- This support is able to satisfy the demands from the applications to a reasonable cost and with an acceptable performance
Key Words: Mobile agents, Multicast.

1. Introducción

Las aplicaciones de agentes móviles que involucran áreas extensas de la Internet y están a su vez constituidas por gran cantidad de agentes, presentan graves problemas de escalabilidad aún no resueltos, respecto a la localización de y a la comunicación entre agentes.

En [7] se han identificado los requerimientos más importantes que deben ser satisfechos por la infraestructura de comunicación en este tipo de aplicaciones. Estos requerimientos se enumeran a continuación, especificando la manera en que pueden ser satisfechos por una implementación del soporte multicast a nivel aplicación, cuya ejecución está a cargo de los servidores de agentes móviles.

1-Cobertura de amplias áreas en la Internet: la infraestructura de comunicación multicast debe cubrir el área de acción de los sistemas de agentes móviles. Esta es la totalidad de la Internet o gran parte de ella, dependiendo de cada aplicación y de la distribución de los servidores de agentes móviles. En la actualidad, el soporte multicast a nivel de red está restringido al área cubierta por el MBONE, requiriendo modificaciones en los routers en el caso de que se necesite extenderla. En nuestra propuesta, las funciones multicast se asignan a los servidores de agentes, lo cual permite desplegar rápidamente y con costos mínimos el soporte en aquellos lugares de la red de interés para la aplicación; sólo es necesario extender la funcionalidad de los servidores incorporando un módulo multicast. Este tipo de soporte multicast a nivel aplicación, cuando es utilizado en redes peer to peer, presenta el problema de la imprevisibilidad de los equipos de los usuarios [4], que pueden ser apagados sin previo aviso, dando lugar a costosos procedimientos de recuperación del árbol de distribución, con la consiguiente carga en la red y deterioro de la calidad del servicio. En nuestro caso, este soporte está implementado en los servidores de agentes, que pueden considerarse equipos tan seguros como los routers que implementan multicast a nivel de red.

2- Tipo de árbol de distribución adaptado al tráfico generado por las aplicaciones: el árbol de distribución multicast es el medio a través del cual los paquetes se distribuyen a los receptores, y está compuesto de nodos (raíz, intermedios y hojas) y vínculos de transmisión, por los cuales fluye la información dirigida al grupo. El tipo de árbol utilizado influye de manera significativa en la performance de las aplicaciones y en la carga generada en la red, y debe ser considerado teniendo en cuenta las características del tráfico producido por las aplicaciones. Han sido definidos varios tipos de árboles, cada uno de ellos con características que responden mejor a cierto tipo de tráfico. Para una clasificación de árboles de distribución, referirse a [6].

La implementación de multicast a nivel aplicación presenta la ventaja de poder construir el tipo de árbol de distribución más adecuado para el tráfico generado, en lugar de tener que utilizar el implementado a nivel de red, común a todos los usuarios y con objetivos propios del nivel de red. En el caso de agentes móviles, debe tenerse en cuenta, para seleccionar el tipo de árbol a utilizar, el tráfico producido por los agentes, la cantidad de posibles emisores, y la dinamicidad del grupo. De acuerdo con el uso esperado de la comunicación multicast en este caso (funciones de management y notificación de ciertos eventos relevantes por parte de los agentes al grupo), la comunicación será esporádica, de poco volumen[1], y originada en múltiples puntos (potencialmente cualquier agente puede enviar información). Esto da lugar a la necesidad de un árbol de distribución multicast bidireccional compartido con raíz en el servidor de origen de la aplicación. La alternativa de árboles no compartidos (uno por cada emisor), produciría una carga no admisible en relación al volumen de datos a enviar, debido a la cantidad de emisores y a los mecanismos de construcción de los árboles. Además, al trasladarse un agente de un nodo a otro, seria necesario construir un nuevo árbol de distribución con raíz en el nuevo servidor. Esto indica que debería utilizarse un árbol compartido, es decir, un único árbol que transporte información de todos los emisores. Esta alternativa reduce significativamente la carga, ya que se tendrá un único árbol, con raíz en el nodo desde el que se lanzo la aplicación, y cuyas hojas irán cambiando a medida que los agentes se desplacen de servidor.

3-Adaptabilidad a grupos altamente dinámicos: Los grupos generados por este tipo de aplicaciones son altamente dinámicos, dependiendo de la movilidad de los agentes. La migración de un agente, da lugar generalmente a una solicitud de integración del grupo (join) por parte del nuevo servidor y un abandono del grupo (leave) por parte del anterior. La gran dinamicidad de los agentes móviles plantea dos problemas a considerar. Por un lado, la carga introducida en la red por los procesos de pertenencia y abandono del grupo debe ser mínima. Por otra parte, estos procesos deben llevarse a cabo en un tiempo adecuado al de permanencia del agente en el nuevo servidor para que no se pierda información. Este ultimo aspecto ha sido tratado en [5], donde la movilidad es considerada una fuente adicional de pérdida de información. En nuestra propuesta se plantea un esquema de construcción del árbol de distribución basado en el recorrido del agente en lugar de la topología de la red. Este proceso, si bien menos eficiente en cuanto a la conformación del árbol de distribución, evita la perdida de paquetes y disminuye las demoras en el proceso de pertenencia[2]. Como alternativa a este método propuesto, presentamos una heurística

[1] Precisamente el objetivo de los agentes móviles es evitar el intercambio de grandes volúmenes de información, trasladándose el proceso al lugar donde residen los datos.

[2] Considerando el tiempo total necesario para que un agente que ha migrado a un nuevo servidor comience a recibir información grupal en modo confiable.

simple que mejora sustancialmente las características del árbol de distribución obtenido a través de la explotación de un conocimiento parcial de la topología de la red en el proceso de construcción de dicho árbol.

4- Confiabilidad. Los usos más comunes de la transmisión grupal en sistemas de agentes móviles, son el envío de mensajes de management con el objeto de que la aplicación pueda controlar a los agentes, el envío de información entre los agentes para coordinar su operación, por ejemplo, un agente indica que ha cumplido con su objetivo, y mensajes para la localización de un agente en particular, a fin de iniciar una conexión con él[3]. En general, esta clase de comunicación requiere transmisión confiable. El problema de ofrecer transmisión multicast confiable no esta aún resuelto. Cuando se utiliza soporte multicast a nivel de red, el agregado de confiabilidad se realiza agregando un nivel adicional, situado sobre el de red, denominado multicast confiable. Este campo está sujeto a investigación, no habiendo resultados satisfactorios que permitan pensar en implementaciones a nivel global, y menos en tipos de tráfico como el que nos ocupa. Dos de los protocolos más destacados son Selective Reliable Multicast (SRM) y Tree Based Reliable Multicast Protocol (TRAM), que han sido diseñados para casos particulares de tipos de comunicación, con diferentes características a las necesarias en el caso de agentes móviles.. Aunque existiera un soporte adecuado en el nivel de multicast confiable, el agregado de este nuevo nivel produciría tiempos de respuesta no compatibles con los requerimientos de estas aplicaciones y un consumo de recursos inadmisible. En nuestra propuesta se propone la implementación del árbol de distribución utilizando conexiones TCP entre los nodos (conexiones entre padres e hijos), lo que permite ofrecer confiabilidad de manera simple, integrando la construcción del árbol de distribución con el soporte para confiabilidad..El uso de cadenas de conexiones TCP débilmente relacionadas se propone en [2], pero considerando la adaptación de las mismas a múltiples receptores heterogéneos; en nuestro caso, este problema no es relevante debido a la baja tasa de transmisión generada por la aplicación. Como mejora, en nuestro caso estas conexiones TCP pueden ser utilizadas para soportar canales pertenecientes a varios árboles de distribución y adicionalmente para intercambio de información de control entre los servidores.

5- Seguridad: La transmisión multicast a nivel de red presenta graves problemas de seguridad. Cualquier equipo puede escuchar las transmisiones, y lo que es aún mas grave, generar paquetes para el grupo, produciendo como mínimo la saturación de los recursos asignados[4].La incorporación de un sistema de comunicación grupal no debe representar una disminución en la seguridad. En nuestra propuesta, contrariamente a lo que ocurre en un sistema multicast a nivel de red, el control de acceso al árbol multicast está manejado por los servidores de agentes, que de acuerdo a la aplicación, pueden implementar las medidas de seguridad adecuadas en cada caso.

6- Cobro por uso de recursos: esta característica es de importancia ya que el mantenimiento de un nodo del árbol multicast implica consumo de recursos por parte de los servidores. La dificultad en cargar a los usuarios por el uso de facilidades multicast a nivel de red ha sido un factor que ha demorado su ofrecimiento por parte de los ISPs. En nuestro caso, el cargo por uso de facilidades multicast se realiza con los mismos mecanismos (si los hay) que los utilizados para cargar a la aplicación por el uso de otros recursos propios de la aplicación en el servidor correspondiente.

7-Flexibilidad en el procesamiento de paquetes: En algunas aplicaciones se requiere un tratamiento especial de los paquetes por parte de los routers, ya sea para mejorar el uso de los recursos o para adaptarse a los requerimientos de la aplicación. Esto no puede ser satisfecho con un soporte multicast a nivel de red, ya que debería agregarse en los routers funcionalidad especial de acuerdo a cada caso. En nuestra propuesta, los servidores de agentes pueden someter a los paquetes a los procesos que resulten necesarios, según los requerimientos de cada aplicación. Un caso típico lo constituye la sumarización de asentimientos en soportes de multicast confiable con asentimientos explícitos.

En resumen, las características más importantes de nuestra propuesta son las siguientes:

- Implementación del soporte de comunicación multicast en los hosts, particularmente en los servidores de agentes móviles involucrados en la aplicación.
- Construcción de un árbol de distribución multicast compartido y bidireccional, con raíz en el servidor en el cual se ha iniciado la aplicación.
- Asignación de direcciones multicast únicas para cada aplicación, basadas en la dirección IP del servidor que actúa como raíz del árbol.

[3] La localización de un agente es resuelta con un tipo de comunicación multicast. Un agente emite un requerimiento de localización multicast que solo será respondido (unicast) a la dirección del solicitante por el agente que esta siendo buscado.

[4] Si bien es posible autenticar emisores a nivel aplicación, con lo cual un receptor no aceptará paquetes provenientes de emisores no autorizados, los paquetes dirigidos al grupo, provenientes de equipos mal intencionados de todas maneras son introducidos en la red, provocando el consumo de recursos y el posible "denial of service".

- Uso de protocolo TCP para mantener los vínculos entre nodos padres e hijos que conforman el árbol de distribución

2. Operación

A continuación se describe la operación de los servidores relacionada con la construcción y el mantenimiento del árbol de distribución, la asignación de direcciones multicast, y la transferencia de datos multicast entre los agentes.

2.1. Selección de la raíz del árbol de distribución y asignación de direcciones multicast

El proceso de construcción del árbol de distribución comienza cuando es iniciada una aplicación, activando el primero de los agentes móviles en el servidor seleccionado. Este servidor se convierte en la raíz del árbol de distribución, adquiriendo responsabilidad en la asignación de direcciones multicast a la aplicación y respecto al mantenimiento de dicho árbol.

Las direcciones multicast definidas para nuestro caso, no son las utilizadas en el soporte multicast standard a nivel de red (direcciones clase D en IP versión 4), sino que se componen de una parte inicial, que consiste en la dirección IP del servidor raíz, más una identificación única local a dicho servidor. De esta manera, las direcciones multicast asignadas por diferentes servidores difieren siempre entre sí, evitando el uso de los costosos protocolos de asignación global de direcciones multicast a nivel de red destinados a evitar colisión de direcciones. En nuestro caso, bastará con implementar un método local a cada servidor, que permita elegir sufijos locales diferentes para cada dirección multicast generada.

2.2 Generación periódica de información por parte del nodo raíz

El servidor origen de la aplicación, además de constituirse en raíz del correspondiente árbol de distribución, adquiere responsabilidad en el mantenimiento del mismo: es el encargado de emitir, a intervalos regulares, mensajes heartbeat (multicast) que se difundirán por el árbol de distribución. Un heartbeat, cuando es originado por el servidor raíz, sólo contiene su identificación y el número de nodos hijos que actualmente tiene. Cada servidor que lo reciba, antes de difundirlo a sus hijos, agregará en él su dirección, su distancia al servidor anterior[5] y su número actual de servidores hijos. De esta manera, cualquier servidor conocerá su camino hacia la raíz del árbol Esta información es utilizada 1-para casos de falla de nodos (lo que obliga a reparar el árbol), o 2-para determinar el servidor que conviene seleccionar como padre, en caso de que un nuevo servidor desee integrar el árbol de distribución[6], o 3-para mejorar el árbol de distribución a través de la reconexión de nodos a servidores más adecuados.

2.3. Modificaciones en el árbol de distribución

Las modificaciones que pueden producirse en el árbol de distribución consideradas en este trabajo[7] son 1-altas de nuevos nodos: cuando un agente móvil arriba a un servidor que no pertenece al árbol de distribución, 2-bajas de nodos ya pertenecientes al árbol: por partida del último agente o desconexión del último nodo hijo, y 3-cambios en la composición de dicho árbol con el objeto de mejorar su performance: cuando el nodo adquiera información adicional respecto a la topología del árbol, que le permita mejorar su conexión al mismo (por ejemplo arribo de un nuevo agente a un nodo ya conectado). Las operaciones a realizar difieren de las llevadas a cabo en multicast a nivel de red, debido al tipo de nodos que integran el árbol de distribución (hosts en lugar de routers), a los requerimientos de la aplicación, y a la información topológica utilizada.

Las altas de nuevos nodos, en casos de uso tradicional de facilidades multicast, se producen debido a que un proceso en un host de la red local solicita agregarse al grupo multicast para recibir la información correspondiente. Como consecuencia, el host notifica a su router local utilizando IGMP, y éste inicia el procedimiento de conexión al árbol de distribución. En este caso, la localización de un nodo ya perteneciente al árbol, a quien conectarse, se realiza en base a información topológica (generalmente derivada de las tablas de ruteo unicast), que corresponde a la métrica utilizada como objetivo a mejorar en la construccion del árbol[8]. Esto implica que no será necesario un ajuste del árbol (que este nuevo nodo cambie de padre, si es notificado de uno mejor) excepto en caso de fallas en nodos que hagan que se requiera un reconstrucción de la totalidad o parte del árbol.

---

[5] Para determinar la distancia desde el servidor anterior hasta el que recibe el paquete, no es necesario ningún procedimiento especial, ya que se utiliza el contador de saltos provisto por IP (TTL).

[6] En el caso del método de construcción del árbol que utiliza el conocimiento de la topología de la red.

[7] Otras modificaciones a considerar en la continuación de este trabajo son las relativas a recuperación de la conectividad ante fallas en nodos o vínculos de transmisión.

[8] Un nodo que requiere agregarse al árbol de distribución buscará el próximo nodo en dirección a la raíz, según su tabla de ruteo. Así recursivamente hasta que encuentra un nodo ya perteneciente al árbol de distribución, al cual se agrega como hijo

En el caso de agentes móviles, la incorporación de un nuevo servidor al árbol de distribución se produce como consecuencia de que un agente se traslada al servidor, y solicita recibir información dirigida al grupo. Nuestra solución propone una alternativa diferente respecto a la información en la cual basarse para construir el árbol de distribución: en lugar de utilizar información topológica como se describió, se utiliza información transportada por el agente, consistente en la lista de servidores que componen la rama del árbol desde la raíz hasta el último nodo visitado por el agente, derivada del último heartbeat recibido por dicho servidor. Esto implica que no se dispone de información topológica que permita construir un árbol eficiente (se sacrifica esta característica para obtener un agregado inmediato al árbol con transmisión confiable, prescindiendo de la existencia de un soporte multicast de red generalizado), y por lo tanto, se deberá mejorar el árbol durante la operación normal, basándose en nuevos agentes que soliciten residir en el nodo o en procedimientos periódicos adicionales.[9]

La desconexión de un servidor del árbol de distribución, se producirá cuando éste no tenga agentes locales ni servidores hijos para el grupo. Estas condiciones se pueden producir como consecuencia de la migración o destrucción de agentes locales y de modificaciones en el árbol de distribución debidas a bajas o cambio de padre por parte de otros servidores dependientes de este servidor. Estas condiciones son equivalentes a la no existencia de procesos interesados en recibir información del grupo en la red local y a la desconexión de los nodos dependientes del router en multicast a nivel de red.

En ambos casos, multicast a nivel de red y nuestra propuesta, el nodo (servidor) envía un mensaje solicitando desconexión (leave) a su padre.

2.4. Transmisión de información multicast

Como fue mencionado, el volumen de datos (en nuestro caso, multicast) a transferir en aplicaciones de agentes móviles es bajo, y fundamentalmente derivado de operaciones de management y notificaciones de un agente al grupo, pudiendo originarse en cualquiera de los agentes. Estas características determinan que el árbol de distribución deba ser bidireccional y compartido. Cuando un agente desea emitir un dato multicast, lo solicita a su servidor actual, quien lo difundirá por todas sus interfaces multicast (independientemente de si el interlocutor en la interfaz es padre o hijo). Para realizar esta difusión, utilizará conexiones TCP que ha establecido en el momento de agregarse al árbol (en su rol de hijo) o posteriormente cuando ha agregado a un servidor hijo (en su rol de padre). De esta manera, se tienen las bases para implementar un multicast confiable de manera simple. Además de difundirlo a los servidores adyacentes, el servidor entregará el mensaje a todos los agentes locales de la aplicación, salvo al originador del mensaje. Un servidor que recibe un paquete multicast originado en otro servidor, lo difundirá de manera similar.

Si bien la demora producida por el proceso de agregado al árbol de distribución (joining) por parte de un servidor es mínima, debe tenerse en cuenta que un agente puede perder información si es recibido un paquete durante el lapso de tiempo comprendido entre su partida de un servidor, y el momento en que el nuevo servidor concrete su pertenencia al arbol de distribución y comience efectivamente a recibir información multicast. Para evitar esta pérdida de datos, se prevé que el servidor del cual parte un agente almacene temporalmente la información dirigida a él (en particular la información multicast), para entregarla posteriormente al nuevo servidor que aloje al agente.

3. Una heurística simple para construir y mejorar el árbol de distribución

Si bien la alternativa descripta para agregarse al árbol de distribución se produce en tiempo mínimo, ya que se utiliza como nodo padre el servidor del cual proviene el agente y el nexo se establece utilizando la conexión TCP creada para el transporte del agente, el árbol de distribución producido resulta sumamente ineficiente (como puede apreciarse en los resultados obtenidos en la simulación), debido a que la relación padre-hijo entre los servidores se establece teniendo en cuenta sólo el recorrido de los agentes y no la topología de la red. A continuación se presenta una heurística simple utilizada para la construcción y eventual modificación del árbol de distribución, que conserva la característica de un agregado en un tiempo aceptable[10] mejorando el árbol de distribución obtenido.

Esta heurística ha sido probada a través de simulación (ver sección 4) produciendo resultados satisfactorios. En la actualidad se está trabajando para ampliar su aplicación y mejorarla.

Cuando un agente arriba a un servidor que aún no forma parte del árbol de distribución, producirá el joining al grupo correspondiente por parte de este nuevo servidor. Como consecuencia de no utilizar un soporte multicast a nivel de red, el nuevo servidor debe basarse sólo en la información transportada por el agente para conectarse al

[9] En la actualidad, la manera de mejorar el árbol de distribución se basa en la llegada de nuevos agentes. Se esta mejorando la heurística descripta a continuación para lograr, a costo razonable, que los nodos puedan mejorar su conexión al árbol en base a información provista por el nodo raiz a intervalos regulares, incluida en los mensajes heartbeat. Esta alternativa deberá ser evaluada en base al consumo de recursos que significa esta mejora y al tiempo que permanezcan los agentes en los servidores.

[10] Debe remarcarse que este agregado al árbol de distribución incluye el equivalente al agregado a nivel de red y el establecimiento de las bases para un multicast confiable.

árbol. Esta información, en su versión más simple, consiste en una referencia al servidor inmediatamente anterior, del cual provino el agente. El nuevo servidor solicitara a éste último su agregado como hijo (join). Si bien esta alternativa produce un agregado rápido, sin necesidad de la invocación a otros mecanismos (por ejemplo a los mecanismos de multicast a nivel de red), es altamente ineficiente, ya que produce árboles de distribución con excesiva superposición de nodos y vínculos de transmisión, dependiendo la calidad de los mismos exclusivamente del recorrido que realicen los agentes.

El procedimiento que se describe a continuación tiene por objeto mejorar la calidad del árbol y acercarla a la obtenida con un protocolo multicast a nivel de red a costo razonable. Según los resultados de las simulciones produce árboles aceptables mejorando sustancialmente a la alternativa anterior.

a-Dados dos nodos, a y b, definimos distancia de red entre ellos - Dr(a,b) - como la cantidad mínima de nodos que es necesario recorrer para llegar de uno al otro. Notar que se supone redes simétricas, por lo tanto Dr(a,b) = Dr(b,a).
b-Dados dos nodos a y b pertenecientes a un árbol de distribución, definimos la distancia multicast entre ellos – Dm(a, b) – como la cantidad de nodos que debe visitarse, siguiendo el recorrido indicado por el árbol de distribución, para llegar desde a hasta b. De la misma manera se supone Dm(a,b) = Dm(b,a).
c-Si el camino desde un nodo j de un árbol de distribución hasta otro nodo k del árbol está compuesto por los nodos intermedios x, y, z,

Dm(j, k) = Dr(j,x) + Dr(x,y) + Dr(y,z) + Dr(z,k) y en particular, al ser z padre de k,

Dm(j,k) = Dm(j,z) + Dr(z,k)

d-En general, para cualquier par de nodos a y b del árbol, se cumplirá:
$\mathrm{Dm(a,b)} \geq \mathrm{Dr(a,b)}$, cumpliéndose la igualdad en el caso en que a y b tengan una relación padre-hijo, debido al método de construcción del árbol propuesto[11].

Suponiendo que el camino producido por el método de construcción del árbol entre el nodo raíz n_0 y un nodo miembro n_i está constituido por los nodos $n_0, n_1, ...n_{i-1}, n_i$, los heartbeats generados periódicamente por el nodo raíz, proporcionarán a todo nodo n_k, $0 < k \leq i$, la siguiente información:

-La distancia de red entre cada par de nodos padre-hijo, $Dr(n_r, n_{r+1})$ para todo r tal que $0 \leq r < k$

-El número de nodos hijos de cada nodo anterior, H_{n_r}, para todo r tal que $0 \leq r < k$

Por otra parte, suponemos que cualquier nodo (en particular n_i) puede determinar la distancia entre él y cualquier otro nodo de la red (en particular los integrantes de la rama del árbol) a través de mecanismos de nivel de red (por ejemplo ICMP echo requests y replys). Por lo tanto se conoce $Dr(n_i, n_k), \forall k, 0 \leq k < i$.

En base a la información anterior, un nodo n_i puede obtener las distancias multicast entre la raíz y sus predecesores n_k,

$$\mathrm{Dm}(\mathrm{n}_0, \mathrm{n}_\mathrm{k}) = \sum_{z=0}^{k-1} \mathrm{Dr}(\mathrm{n}_\mathrm{z}, \mathrm{n}_{\mathrm{z+1}})$$

De esta manera, un nodo k que ha recibido un agente y debe conectarse al árbol de distribución, seleccionará el padre más adecuado contando con la información proporcionada por el agente, que a su vez la ha obtenido del último servidor visitado, actualizada al último heartbeat recibido. El nodo padre n_p será seleccionado, según nuestra heurística, en base a tres aspectos :

1- la distancia multicast $\mathrm{Dm}_{\mathrm{n_p}}$ que lo separara del nodo raíz del árbol en caso de utilizar como padre a n_p, $\mathrm{Dm}_{\mathrm{n_p}}(n_0, n_k) = \mathrm{Dm}(n_0, n_p) + Dr(n_k . n_p)$. El hecho de hacer mínima esta distancia, tiende a que el nodo se conecte al árbol de distribución en un punto que no produzca demasiada carga debido a involucrar superposición de vínculos a nivel de red. Sin embargo, teniendo en cuenta solo la minimización de este parámetro, obtendríamos un árbol compuesto de tantas ramas nodo_raíz–nodo_miembro como miembros del árbol haya, ya que los nodos tenderían a conectarse directamente al nodo raíz. Como consecuencia, tendríamos sólo vínculos punto a punto[12] con la consiguiente sobrecarga del servidor que actúa como nodo raíz y sus adyacencias.

[11] La igualdad puede cumplirse en otros casos especiales, dependiendo de la ubicación de los nodos en la red.
[12] Sólo se limitaría la conexión al nodo raíz en función del peso asignado a la cantidad de hijos de un servidor.

2- la distancia de red entre todos los posibles padres y el nodo en cuestión. Hacer mínima esta distancia tiende a lograr árboles de distribución en los que los nodos adyacentes se encuentren cercanos entre sí, lo cual mejora la carga global. Teniendo en cuenta sólo la minimización de esta distancia, se obtendría un comportamiento similar al que tienen los protocolos multicast a nivel de red con pertenencia explícita, pero en este caso trabajando con un conjunto restringido de posibles nodos padre (solo los servidores de la rama a través de la cual arribó el agente).
3- la cantidad de nodos hijos que ya tiene un posible candidato a ser padre. Este factor se tuvo en cuenta para evitar una carga excesiva en los servidores, y así obtener un árbol con un número limitado de hijos por servidor. Si bien aquí se tuvo en cuenta sólo la cantidad de hijos, podría reemplazarse este valor por otros relacionados con el costo económico de establecer una bifurcación en un servidor, con motivos de seguridad, etc. En definitiva, este valor puede ser determinado por cada aplicación.

Según la heurística desarrollada, para un nodo Z, perteneciente al árbol de distribución con raíz en R, el costo C_{p_i} para cada potencial padre p_i, se calcula de la siguiente manera:

$$C_{p_i} = a * Dm_{p_i}(R,Z) + (1-a) * Dr(p_i,Z) + b * H_{p_i},$$

donde Dm_{p_i} es la distancia multicast entre la raíz R y el nodo Z, si éste utiliza como padre a p_i, y H_{p_i} es el número de hijos del nodo p_i, y ($0 \le a \le 1$) y b son valores arbitrarios a seleccionar; a determina la importancia relativa dada a los puntos 1 y 2, mientras que b representa el costo arbitrario asociado a cada nodo según el número de hijos que tenga en caso de ser seleccionado como padre.

Para determinar el nodo al cual se solicitará el join, se establece un orden considerando todos aquellos nodos de los cuales se dispone de información, es decir, los integrantes de la rama por la cual arribó el agente. El nodo intentará un join con el posible padre de costo mínimo, y en caso de no recibir respuesta debido a fallas, a que dicho nodo ya no pertenece al árbol de distribución ya que la información está actualizada al último heartbeat, o debido a que tiene colmada su capacidad, intentará con los demás en orden ascendente. Para minimizar el tiempo de joining en caso de respuestas negativas, el nodo podría enviar múltiples joins simultáneos, con el consiguiente aumento de carga en la red.

Además de utilizar esta heurística para que un nodo no conectado al árbol seleccione un padre adecuado, se propone su uso en otros casos, por ejemplo cuando un servidor ya integrante del árbol recibe un agente, puede recalcular los costos al conjunto ampliado de posibles padres, los propios más los pertenecientes a la rama del árbol a través de la cual arribó el agente.. En caso de detectar un nodo en mejores condiciones que el actual padre, se producirá el correspondiente cambio, siendo esto transparente a los servidores descendientes de los nodos involucrados.

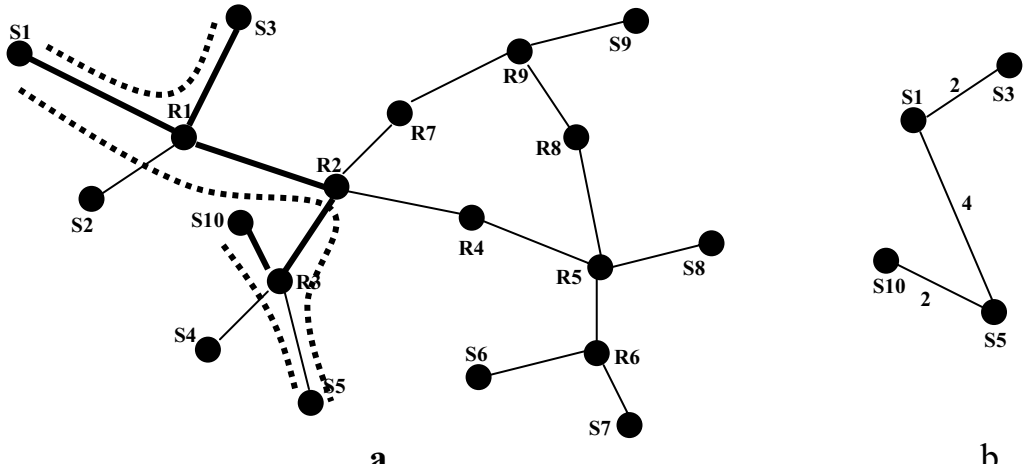


Fig.1 a-Ärboles de distribución obtenidos por un protocolo multicast a nivel de red líneas gruesas de router a router) y utilizando la heurística descripta (líneas punteadas de servidor a servidor). Las líneas punteadas se han dibujado de manera tal que reflejen los routers involucrados en el trafico de paquetes. b- Arbol de distribución a nivel servidores (entre paréntesis se indica la distancia en nodos).

Este mismo chequeo de padres podrá hacerse en el caso de detectar un cambio en la información recibida a través de los heartbeats. Debe tenerse en cuenta que no existe el riesgo de producir ciclos en el árbol, debido a que un nodo que selecciona un padre, conoce la secuencia de nodos entre él y el raíz. Otro caso de recálculo de costos

puede darse si se contemplan procedimientos para difundir información respecto a la conformación del árbol a los nodos (además de los heartbeats, que proporcionan información respecto a la rama a la cual pertenece cada servidor).

En la figura 1 se muestra el árbol de distribución construido con la heurística descripta, comparándolo con un árbol construido por un protocolo multicast a nivel de red, suponiendo el mismo orden de solicitudes de joining por parte de los nodos.

3.1 Ejemplo de funcionamiento

A continuación se describe el comportamiento de los servidores de agentes al recibir un agente y agregarse, en consecuencia, al árbol de distribución. La topología de la red y la ubicación de los servidores corresponde a la figura 1. Los parámetros utilizados en la función son a = 0,75 y b = 4.

1-El agente **a** es activado en el servidor **S1**, como consecuencia éste se convierte en raíz del árbol de distribución
2-Agente **a** se traslada a **S5**, lleva la información (S1,0,0) que hace referencia al único servidor conocido por el agente; en este caso la distancia de red al servidor anterior carece de sentido (se le da valor cero) y el número de hijos del servidor, es cero (no tiene servidores hijos).
3-Servidor **S5** realiza el join con **S1**, único servidor conocido. El camino a nivel de red es S5,R3,R2,R1,S1
4-Agente **a** se clona en **S5**, produciendo agente **b**. Agente **a** migra a **S3**, y **b** migra a **S9**
5-**S3** recibe de agente **a** la información (S1,0,1/S5,4,0), indicando el camino desde la raiz recorrido por el agente (**S1** y **S4**), la distancia de red entre **S4** y **S1** (4), y el número de hijos de ambos servidores. De la aplicación de la función definida por la heurística se obtiene:
C(S1) = 0,75 * 2 + 0,25 * 2 + 4 * 1 = 6
C(S5) = 0,75 * 8 + 0,25 * 4 + 4 * 0 = 7
Por lo tanto decide join con **S1**, el servidor de menor costo
6- **S9** recibe de agente **b** la información (S1,0,1/S5,4,0), aplica la función y obtiene:
C(S1) = 0,75 * 5 + 0,25 * 5 + 4 * 1 = 9
C(S5) = 0,75 * 9 + 0,25 * 5 + 4 * 0 = 8
Por lo tanto decide join con **S5**
7-Luego de los correspondientes joins, suponemos que **S1** emite su heartbeat: (S1,0,2), que indica identificación del servidor, distancia al servidor anterior en el árbol de distribución y número de hijos. Es recibido por **S3** y **S5**
S5 lo actualiza con su información: (S1,0,2/S5,4,1) y lo envía a **S9**
8- Agente **b** en **S9** migra a **S10**, éste recibe la información (S1,0,2/S5,4,1/S9,5,0). Calcula costos y obtiene:
C(S1) = 0,75 * 0 + 0,25 * 4 + 4 * 2 = 9
C(S5) = 0,75 * 4 + 0,25 * 2 + 4 * 1 = 7,50
C(S9) = 0,75 * 9 + 0,25 * 5 + 4 * 0 = 8
S10 decide por lo tanto conectarse vía **S5**, y como consecuencia **S9** se desconecta del árbol.
9- El agente **b** en **S10** se clona produciendo un agente **c** que decide migrar a **S3**
10- **S1** envía un nuevo heartbeat, (S1,0,2).
11-**S5** lo actualiza con su información y lo envía a **S10**, (S1,0,2/S5,4,1)
12-**S3** recibe al agente **c**, con la información proveniente de **S10**: (S1,0,2/S5,4,1/S10,2,0).
Además de esta información, el nodo **S3** dispone de la suya: (S1,0,2) correspondiente al último heartbeat generado por el nodo raíz **S1** (en este caso no agrega ningún nodo a los ya conocidos). Aunque **S3** ya pertenece al árbol de distribución, chequeará nuevamente los costos por si logra reducir el actual:
C(S1) = 0,75 * 0 + 0,25 * 2 + 4 * 2 = 8,5 – 4 = 4,5 (el propio)
C(S1) = 0,75 * 0 + 0,25 * 2 + 4 * 2 = 8,5 – 4 = 4,5 (recibido de S10)
C(S5) = 0,75 * 4 + 0,25 * 4 + 4 * 1 = 8
C(S10) = 0,75 * 6 + 0,25 * 4 + 4 * 0 = 5,5

Debe tenerse en cuenta que cuando no se trata del nodo raíz, la evaluación de datos para un posible nodo padre, recibidos de diversas fuentes puede dar lugar a obtener distintos costos, ya que las distancias multicast al nodo raíz pueden diferir. Al estar **S3** conectado directamente como hijo a **S1**, esto debe reflejarse en la función de costos, restando uno al número de hijos anunciado por **S1**. De esta forma, **S3** decide no cambiar de padre. Notar que en las mismas circunstancias pero con **S3** no miembro del árbol, se hubiera decidido por el join a **S10**.

4. Simulación

Con el objeto de estimar la performance de nuestra propuesta, se realizaron simulaciones para las siguientes alternativas: 1-para un protocolo a nivel de red que construye árboles bidireccionales alternativos, Core Based Trees [1], tomado como referencia; 2-para la versión más simple de soporte multicast implementado a nivel servidores de agentes, en la cual un servidor se conecta al árbol de distribución utilizando el último servidor visitado por el agente que solicita pertenencia al grupo, y 3-para multicast implementado a nivel servidores de agentes utilizando la heurística descripta en la sección 3.

Las simulaciones fueron realizadas utilizando el simulador de redes Network Simulator (Ns) [3]. Se desarrolló el código (simplificado) para soporte multicast en los routers en el caso de CBT y el necesario en los servidores de agentes para las dos alternativas restantes. La topología utilizada consistió de 256 nodos interconectados por 266 vínculos de transmisión y se generó a partir de una pequeña porción real de la Internet.

Cada corrida de la simulación consistió en ejecutar una aplicación inicializada en un servidor elegido al azar, compuesta al comienzo por un único agente. La movilidad y frecuencia de clonado de los agentes se generó al azar, no correspondiendo con ningún patrón observado para algún tipo de aplicaciones en particular, análisis que se realizará en el futuro. De esta manera, cada agente perteneciente a la aplicación permanece un tiempo en un servidor, pudiendo luego generar clones y seleccionar al azar un nuevo servidor para migrar. Estos clones se comportan de la misma manera que el agente original.

Para no generar una cantidad de agentes excesiva para la red definida, su máximo fue restringido a 10, luego de alcanzarse esta cantidad, los agentes pueden migrar, pero no clonarse. La cantidad máxima de migraciones se configuró en 80, dado que la visita de varios agentes a un nodo no se correspondería con la realidad en redes de área extensa.

Las corridas consistieron en la generación y migración de agentes hasta alcanzar los números máximos mencionados. Luego, se enviaron datos desde uno de los agentes al resto, con el objeto de evaluar la performance del árbol de distribución obtenido en cada caso. Los resultados obtenidos, para las tres alternativas, se muestran en las figuras 2, 3 y 4.

Los parámetros evaluados fueron los siguientes:

1-Relativos a la calidad del árbol de distribución obtenido:

a-Carga en número de paquetes producidos en la red como consecuencia del envío de un paquete de datos multicast dirigido al grupo por parte de un agente. b-Tiempo promedio de las demoras registradas por los agente en recibir dicho paquete. Estos valores nos dan una idea de la medida en que el árbol de distribución se ajusta a la topología de la red. Como es obvio, el protocolo multicast a nivel de red produce los mejores resultados, tanto en tiempos de envío como en carga en la red, ya que explota la topología. En los casos de multicast implementado a nivel aplicación, la diferencia entre la solución más simple, que sólo tiene en cuenta la secuencia de migración de los agentes y la que evalúa padres alternativos, que explota un conocimiento parcial de la topología de la red y está limitada respecto a los nodos que pueden integrar el árbol de distribución (sólo servidores de agentes), es significativa, mejorándola en ambos aspectos aproximadamente en un factor de 3.

2-Relativos a los mecanismos de pertenencia al árbol de distribución.

Se evaluó: a-la carga producida por las solicitudes de pertenencia al árbol (joins), su confirmación y b-los mensajes de abandono del árbol (leaves) y la demora de los nodos para lograr su pertenencia al árbol de distribución. En ambos casos se observó que la solución a nivel de red mejora a las soluciones a nivel aplicación. Esto se debe a que el soporte multicast a nivel de red sólo provee conectividad entre los miembros del grupo, sin ofrecer ningún tipo de soporte adicional para transmisión confiable, como lo hacen las soluciones a nivel aplicación. Entre éstas, el agregado de miembros al árbol acorde a la secuencia de migración de los agentes resulta mejor, debido a que no es necesario buscar a ni establecer una nueva conexión TCP con el servidor padre, que es aquél del cual provino el agente. Sin embargo, esta aparente mejora debe confrontarse con la baja calidad del árbol de distribución obtenido.

3-Relativos a tiempos de migración

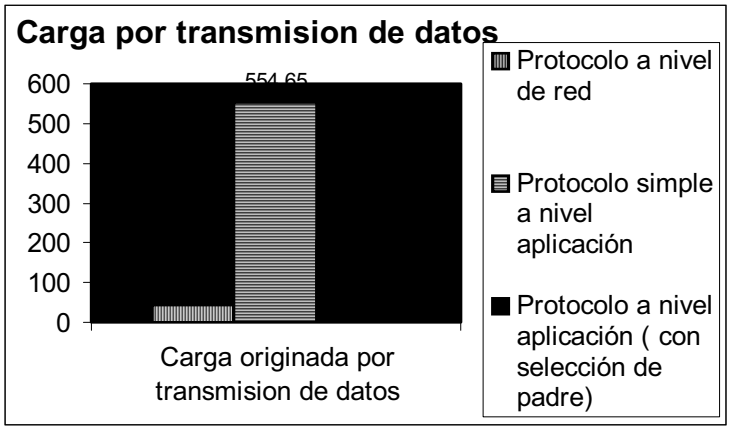
El tiempo de migración de un agente, es tomado desde que el agente solicita migrar a su servidor actual, hasta que es activado en el nuevo servidor, estando éste en condiciones de recibir información multicast. Por lo tanto, incluye el tiempo de transferencia del agente, y el posible tiempo de incorporación al árbol de distribución por parte del nuevo servidor. Cabe aclarar que también en este caso, aunque el tiempo obtenido por el protocolo multicast a nivel de red es menor, no incluye soporte alguno para posibilitar transmisión confiable de datos. La incorporación de un mecanismno de este tipo aumentaría considerablemente los tiempos obtenidos. En el caso de la solución basada en el recorrido de los agentes, el tiempo es menor debido a que no es necesario localizar y conectarse a un nuevo servidor de agentes, ya que el padre del nuevo servidor es el servidor anterior.

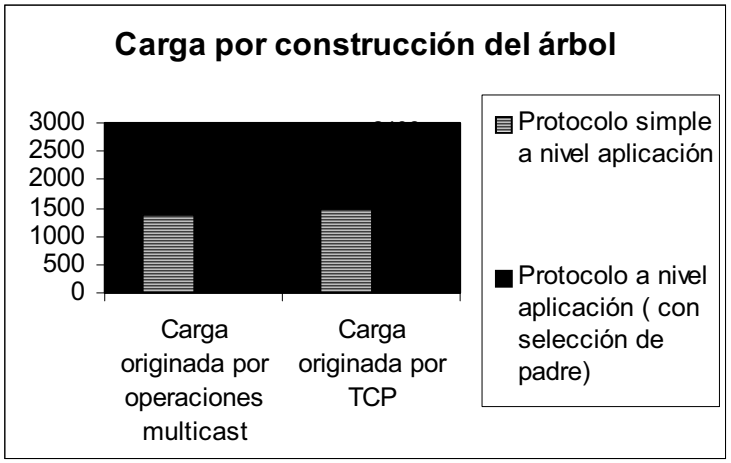
4-Carga a nivel TCP

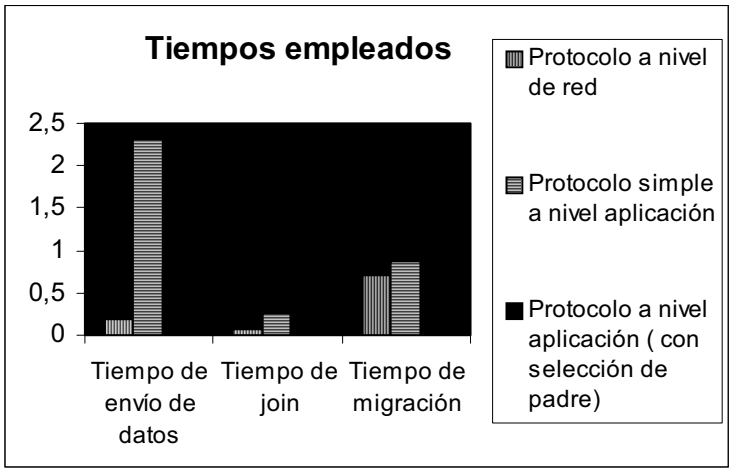
Este parámetro fue evaluado sólo en los casos de multicast implementado a nivel aplicación. Se evaluó la cantidad de paquetes introducida en la red como consecuencia de las conexiones TCP necesarias para lograr la transferencia de los agentes y la construcción del árbol de distribución, implementado como sucesiones de conexiones TCP compartidas.

Nuevamente, en el caso del árbol construido de acuerdo al itinerario de los agentes, se obtiene una mejora debido a que se comparte la conexión TCP entre los servidores involucrados parar el envío del agente y para las operaciones de pertenencia al árbol y posterior envío de datos.

Según los resultados obtenidos, puede concluirse que si bien una solución multicast a nivel de red presenta mejor performance en todos los aspectos, no tiene en cuenta el ofrecer un soporte de transmisión confiable, necesario para el trafico a intercambiar entre los agentes móviles. La incorporación de un nivel de multicast confiable, daría lugar a tiempos y cargas que superarían a los obtenidos por las soluciones de multicast a nivel







Figs. 2, 3 y 4. Resultados obtenidos de las simulaciones

aplicación[13]. Otro inconveniente lo constituye la no existencia de tal soporte para grupos altamente dinámicos que se extienden en áreas extensas de la Internet. Por otro lado, el uso de soluciones a nivel de red no permite desplegar los servicios requeridos en aquellas zonas de la red sin soporte multicast.

Si bien la solución más simple con multicast a nivel aplicación aparenta ser más liviana en cuanto a la carga y más rápida respecto a los tiempos registrados para los procesos de join y migración, se considera que estas diferencias se ven ampliamente superadas por la baja calidad del árbol de distribución obtenido, lo que se refleja en las cargas y tiempos relativos al envío de los datos.

5. Conclusiones y trabajo en curso

La solución propuesta al problema de comunicación grupal y localización en sistemas de múltiples agentes móviles operando sobre áreas extensas, de acuerdo a los resultados obtenidos, puede ser considerada una alternativa válida al uso de multicast a nivel de red. Las ventajas de mayor peso de nuestra solución sobre un soporte multicast tradicional, radican en la posibilidad de desplegar el sistema sin necesidad de una infraestructura multicast en la red, y la sencillez de implementación. Otra característica a destacar es que nuestra solución, al utilizar los servidores como nodos del árbol de distribución, y vincularlos a través de conexiones TCP, allana el camino para una implementación de transmisión grupal confiable, característica que con el esquema tradicional requiere la implementación de un nivel adicional, multicast confiable, el cual aún no ha sido definido satisfactoriamente para el tipo de características de comunicación que presentan los sistemas de agentes móviles.

De acuerdo a los resultados obtenidos en las simulaciones realizadas, considerando tiempos y carga en la red, esta solución resulta factible.

En la actualidad se está trabajando en la mejora de la heurística propuesta, mediante el ajuste de los parámetros definidos y con el agregado de métodos adicionales que permitan aplicarla de manera regular (sin necesidad de la visita de nuevos agentes) para mejorar el árbol de distribución. Debe ser evaluado cuidadosamente el costo de esta alternativa, confrontándolo con el beneficio obtenido de mejorar el árbol, ya que éste puede verse disminuido considerablemente debido a la movilidad de los agentes que puede ocasionar que un servidor permanezca en el grupo multicast por un período breve de tiempo.

Otro aspecto en el que se comenzará a trabajar es la implementación de distintas alternativas de transmisión confiable a ofrecer sobre el soporte definido (asentimientos implícitos, explícitos, etc.) previo relevamiento de las necesidades de comunicación grupal confiable de este tipo de aplicaciones,

Paralelamente se implementará el soporte propuesto en Java con el objeto de comprobar su comportamiento en un escenario real (aunque restringido en el número de servidores) a través del desarrollo de una aplicación simple.

Referencias

[1], "Core based trees (CBT)" T. Ballardie, P. Francis & J. Crowcroft, SIGCOMM Symposium on Communications Architectures and Protocols , pp. 85-95, ACM, September 1993..

[2] "Reliable Overlay Multicast with Loosely Coupled TCP Connections –ROMA", Gu-In –Kwon, John W. Byers, Proc. of IEEE INFOCOM '04, Hong Kong, March 2004.

[3] K. Fall (ed), "Ns Notes and Documentation", VINT Project, UC Berkeley, LBL, USC/ISI, Xerox PARC, March 2000.

[4], "Transcience of Peers and Streaming Media", Mayank Bava, Hrishikesh Deshpande, Hector Garcia-Molina, ACM SIGCOMM Computer Communication, Volume 33 , Issue 1, January 2003 Pp: 107 - 112 [Murphy,2002]

[5]"Reliable Communications for Highly Mobile Agents", Amy Murphy, Gian Pietro Picco, Journal of Autonomous Agents and Multi-Agent Systems, vol. 5, no. 1, pp. 81-100, March 2002.

[6] "Intra- and Inter- domain multicast routing protocols: A survey and taxonomy", M. Ramalho, IEEE Communications Surveys and Tutorials, vol.3, no.1, pp.2-25, Jan.-Mar. 2000

[7] "Infraestructura de comunicación para sistemas de agentes móviles". Rigotti, G, Proceedings AST 2003, 32 JAIIO (Jornadas Argentinas de Informática e Investigación Operativa), Buenos Aires, Argentina, September 2003.

[13] El overhead producido por el nivel de multicast confiable es estimativo, ya que su evaluacion requeriria el diseño e implementacion de un protocolo.

Optimización del Tiempo de Ejecución en Problemas de Dinámica Molecular

Angela Di Serio
Universidad Simón Bolívar, Departamento de Computación y TI,
Caracas, Venezuela, 1080-A
adiserio@ldc.usb.ve

and

María B. Ibáñez
Universidad Simón Bolívar, Departamento de Computación y TI,
Caracas, Venezuela, 1080-A
ibanez@ldc.usb.ve

Abstract

Molecular Dynamics (MD) is a powerful tool used to study the properties of molecular systems and their interactions. MD simulations are computational intensive that requires run on parallel architectures in order to produce results in reasonable time. Because of their dynamic nature, the processors workload change along the simulation. In order to minimize the execution time, the processor workloads need to be reassigned. Recent research has showed that the Generalized Dimension Exchange algorithm improves the total execution time of MD simulation distributing molecules uniformly between processors. Nevertheless, processors can consume different amount of time to execute the balanced workload. In this work, we present a different approach to distribute the workload between the processors based on the execution time. The experiments performed show that the total execution time of the simulation is reduced.
Keywords: Dynamic Load Balancing, Distributed Load Balancing, Parallel Applications, Short Range Molecular Dynamics.

Resumen

Dinámica Molecular (DM) es una herramienta de gran utilidad para el estudio de las propiedades de los sistemas moleculares y de sus interacciones. Es una aplicación de cómputo intensivo que requiere ser ejecutada en arquitecturas paralelas para producir resultados en tiempos razonables. Debido a su naturaleza dinámica, la carga de trabajo de los procesadores cambia a lo largo de la simulación. Por lo tanto, para lograr minimizar el tiempo de ejecución es necesario redistribuir la carga entre los procesadores. Investigaciones recientes han mostrado que el uso del algoritmo Generalized Dimension Exchange mejora el tiempo total de ejecución de DM distribuyendo la carga uniformemente entre los procesadores. Sin embargo, los procesadores pueden consumir cantidades de tiempo diferentes para ejecutar la carga balanceada. En este trabajo presentamos una alternativa diferente para distribuir la carga entre los procesadores basado en el tiempo de ejecución. Los experimentos realizados muestran que la mejora logró reducir el tiempo de ejecución de la simulación de DM.
Palabras claves: Balance de Carga Dinámico, Balance de Carga Distribuido, Aplicaciones Paralelas, Dinámica Molecular de corto alcance.

1. INTRODUCCIÓN

La simulación de Dinámica Molecular (DM) es una técnica de gran utilidad para el estudio de las propiedades de los sistemas moleculares y de sus interacciones. Se utiliza para simular las propiedades de líquidos, sólidos [5],[6]. Es una aplicación de cómputo intensivo que requiere ser ejecutada en arquitecturas paralelas para producir resultados en tiempos razonables.

La simulación consiste en calcular la interacción entre las moléculas de un sistema, afortunadamente las interacciones son independientes y pueden ser calculadas en paralelo, pero generan serios problemas de balance de carga. Inicialmente todos los procesadores tienen la misma cantidad de moléculas. A medida que avanza la simulación, las moléculas cambian de posición y por lo tanto, la carga de trabajo de los procesadores se modifica. La paralelización del algoritmo se vuelve ineficiente después de cierto tiempo. Para poder minimizar el tiempo de ejecución, es necesario mantener balanceada la carga de trabajo de todos los procesadores que participan en la simulación.

En los últimos años se han desarrollado diversos algoritmos de balance de carga para distintas clases de aplicaciones científicas [9],[10],[11]. Las técnicas de balance de carga pueden ser clasificadas en estáticas y dinámicas. Un algoritmo de balance de carga estático asigna la carga a los procesadores durante la compilación. Esta técnica no puede ser aplicada a los problemas de DM por su naturaleza dinámica. Los métodos de balance de carga dinámicos distribuyen la carga durante la ejecución basado en el comportamiento de la aplicación. Uno de los métodos más populares de redistribución de carga para problemas de DM es el método recursivo de bisección geométrico [4],[7]. Sin embargo, este método requiere sincronización global y transferencia intensiva de datos entre los procesadores que resulta en una pobre escalabilidad y altos costos computacionales [9]. Existen técnicas centralizadas y distribuidas para la toma de decisiones, y la migración de la carga puede considerar el dominio global de procesadores o un dominio reducido como por ejemplo el conjunto de los vecinos más cercanos.

Trabajos anteriores [2], [3], nos permitieron concluir que el método de balance de carga dinámico distribuido conocido como Generalized Dimension Exchange (GDE) logra mejoras en el tiempo total de ejecución de la simulación de DM distribuyendo uniformemente el número de moléculas entre los procesadores. Sin embargo, se ha observado que procesadores con igual número de moléculas consumen diferentes cantidades de tiempo en realizar la simulación y en ocasiones la diferencia en tiempo es significativa. En este trabajo se presentan mejoras al balanceador basado en GDE que corrigen este comportamiento de los procesadores en nuestra aplicación.

El trabajo se encuentra organizado de la siguiente manera. En la sección 2, se describe en qué consiste la simulación de Dinámica Molecular. En la sección 3, se presentan los componentes de un balanceador de carga genérico y las decisiones específicas que se toman para resolver el problema planteado. Los experimentos realizados se muestran en la sección 4 y finalmente, la sección 5 contiene las conclusiones.

2. SIMULACIÓN DE DINÁMICA MOLECULAR

Dinámica Molecular es una herramienta computacional usada para simular las propiedades de líquidos y sólidos. Cada una de las *N* moléculas en la simulación es tratada como una masa puntual y las ecuaciones de Newton son integradas para calcular su movimiento. Diversas técnicas han sido desarrolladas para paralelizar simulaciones de DM de corto alcance. La que mejor se adapta a la aplicación es la descomposición espacial de Plimpton [8] en donde cada procesador es responsable de simular una porción del dominio del sistema. Para ello, el espacio de simulación es dividido en pequeñas cajas de tres dimensiones y cada una de estas cajas es asignada a un procesador.

Durante la evolución de la simulación, las moléculas se mueven y pueden abandonar el espacio del procesador y entrar en el espacio de otro. Las moléculas son reasignadas a los nuevos procesadores a medida que se desplazan por el espacio de simulación. Para poder calcular las fuerzas sobre las moléculas, es necesario que lo procesadores conozcan las posiciones de las moléculas que se encuentran en los procesadores vecinos. El tamaño de las cajas asignadas a cada procesador dependerá del número de moléculas y del número de procesadores que se usen en la simulación.

En el algoritmo de descomposición espacial , cada procesador mantiene una lista con las moléculas asignadas y otra con las moléculas que se encuentran en los procesadores vecinos y que están dentro de un cierto alcance. Una iteración del algortimo incluye las siguientes actividades:

1. Construcción de la lista de moléculas que salieron del espacio del procesador y envío a sus vecinos. A su vez, el procesador recibe la lista de moléculas que salieron de otros procesadores y entraron a este espacio. Esto se lleva a cabo cada 20 iteraciones
2. Construcción de la lista de moléculas vecinas
3. Cálculo de las fuerzas de las moléculas usando la lista de vecinos
4. Actualización de las posiciones de las moléculas

5. Comunicación de las nuevas posiciones de las moléculas

El problema que surge con este tipo de paralelización es que la simulación puede volverse ineficiente después de un cierto tiempo, debido al desplazamiento de las moléculas. La ineficiencia puede ser corregida redistribuyendo la carga entre los procesadores, i.e. aumentando o disminuyendo el espacio asignado a los procesadores.

3. BALANCE DE CARGA DINÁMICO

El objetivo de los algoritmos de balance de carga es mantener la carga de trabajo de los procesadores aproximadamente igual. Para lograr este objetivo es importante identificar cuatro componentes que usualmente conforman un balance de carga dinámico:

- Regla de Medida de Carga. Los algoritmos de balance de carga dinámico se basan en información relacionada con la carga de los procesadores. Típicamente se caracteriza mediante un índice de carga que suele ser un entero no negativo cuyo valor es cero si el procesador está libre, y que toma valores mayores que cero a medida que el procesador tiene una carga mayor.

- Regla de Intercambio de Información. Especifica el instante de tiempo en el cual se llevará a cabo la recolección y el mantenimiento de la información de la carga de trabajo de los distintos procesadores.

- Regla de Inicio de Balance. Establece cuándo se debe iniciar un proceso de balance de carga. La regla de inicio debería tomar en cuenta el costo de la operación de balance versus el beneficio en cuanto a rendimiento que se obtiene una vez realizado el balance.

- Operación de balance. Puede ser definida mediante tres reglas: regla de localización, regla de distribución y regla de selección. La regla de localización determina el grupo de procesadores o dominio de balance que intervendrán en la operación de balance con respecto a un procesador dado. La regla de distribución determina cómo se redistribuye la carga entre los procesadores que se encuentran en el dominio de balance. La regla de selección determina la carga más conveniente que será migrada entre los procesadores.

3.1 REGLA DE MEDIDA DE CARGA

La cantidad de trabajo que realiza un procesador está en relación directa con el número de moléculas que pertenecen a su espacio de simulación, es por ello que la medida natural de carga sea el número de moléculas que posee un procesador. Esta medida de carga ha sido utilizada con buenos resultados en [2] y [3] pero no considera los efectos de la aglomeración de moléculas en términos de cantidad de trabajo a realizar. El incremento en el número de moléculas en un área hace que el número de vecinos de las moléculas de esa zona crezca , aumentando exponencialmente la cantidad de trabajo. Moléculas con un mayor número de vecinos ocasionan un mayor volumen de trabajo. En este sentido, parece más conveniente usar medidas de carga ligadas al tiempo. Dado que la simulación se hace a intervalos de 20 iteraciones, utilizaremos como medida de carga el tiempo que el procesador se demora en ejecutar las últimas 20 iteraciones de la simulación (T_{actual}) dado el número de moléculas presentes en el espacio de dicho proceso.

3.2 REGLA DE INTERCAMBIO DE INFORMACION

La carga de trabajo de los distintos procesadores será reportada cada vez que la simulación recalcula la lista de moléculas vecinas, es decir, cada 20 iteraciones.

3.3 REGLA DE INICIO DE BALANCE

La métrica usada para decidir si el sistema está balanceado o no es el Coeficiente de Variación (COV) de los tiempos de ejecución de las últimas 20 iteraciones de la simulación. Esta métrica permite escoger el nivel de desbalance en el cual será activado la operación propiamente dicha de balance de carga. Cuando el coeficiente de variación de los tiempos exceda el valor prefijado entonces se activará la operación de balance.

3.4 OPERACION DE BALANCE

Para llevar a cabo la operación de balance propiamente dicha, se escogió el método Generalized Dimension Exchange (GDE) [11]. GDE permite calcular el flujo de carga que deberá migrar entre los distintos nodos para alcanzar un balance de carga. El método GDE puede ser clasificado como un algoritmo de balance de carga determinístico y que sólo permite comunicaciones con los vecinos. Cada procesador compara su carga con la de sus vecinos uno tras otro. En cada una de estas comparaciones, el procesador trata de igualar su carga con la de sus vecinos. Esta operación es repetida hasta que la información de la carga de trabajo se propaga hasta todos los nodos de la red. En este punto surgen dos interrogantes: cuánta carga de trabajo debe recibir o ceder un procesador y cuántas veces se debe intercambiar la información para propagar la información hasta todos los nodos de la red. Esto dependerá de la topología escogida. En nuestro caso se escogió una topología tipo cadena. El flujo de carga

entre dos procesadores vecinos será igual a la cantidad de moléculas que debe migrar de forma tal que el tiempo necesario para procesar la nueva carga sea aproximadamente igual en los dos procesadores involucrados. Este proceso de intercambio y de igualación de carga entre los procesadores se repite *(diametro de la cadena / 2)* veces [11].

El cálculo del flujo de carga a migrar entre los distintos nodos puede ser descrito de la siguiente forma:
Sea

- P_i el procesador i en donde se ejecuta la aplicación ($0 = i = n-1$)
- $T_{actual}(P_i, j)$ Tiempo de ejecución en el procesador *i* del *j*-ésimo grupo de 20 iteraciones
- $M(P_i, j)$ Número de moléculas asignadas al procesador *i* en el *j*-ésimo grupo de 20 iteraciones
- *Moléculas* Número total de moléculas

Lo que se desea es que en un intervalo cualquiera de iteraciones se cumpla que

$$T_{actual}(P_0, j) \cong T_{actual}(P_1, j) \cong \ldots \cong T_{actual}(P_{n-1}, j)$$

De esta forma se garantiza que todos los nodos ejecuten de forma sincronizada y que finalizarán su ejecución aproximadamente al mismo tiempo. El problema que deseamos resolver puede reformularse de la siguiente forma para el j-ésimo grupo de 20 iteraciones

$$\textit{Minimizar } y = \textit{máximo } \{ \; M(P_0, j) * T_{actual}(P_0, j-1) / M(P_0, j-1) \;,$$

$$M(P_1, j) * T_{actual}(P_1, j-1) / M(P_1, j-1) \;, \ldots,$$

$$M(P_{n-1}, j) * T_{actual}(P_{n-1}, j-1) / M(P_{n-1}, j-1) \}$$

Sujeto a:

$$M(P_0, j) + M(P_1, j) + \ldots + M(P_{n-1}, j) = \textit{Moléculas}$$

$$M(P_i, j) = 0 \;\; \forall i \quad \text{y} \quad M(P_i, j) \text{ entero}$$

donde $T_{actual}(P_i, j-1) / M(P_i, j-1)$ es un aproximado de lo que costó ejecutar el grupo anterior de 20 iteraciones para una molécula en el nodo P_i.

Este problema puede ser resuelto usando GDE, en donde en cada iteración *m* del algoritmo se resuelve el siguiente problema:

Para dos nodos vecinos P_i y P_k se desea que

$$M(P_i, j) * T_{actual}(P_i, j-1) / M(P_i, j-1) = M(P_k, j) * T_{actual}(P_k, j-1) / M(P_k, j-1) \quad \text{y}$$

$$M(P_i, j) + M(P_k, j) = M(P_i, j-1) + M(P_k, j-1)$$

Al resolver estos sistemas de ecuaciones obtenemos la carga que debería tener cada uno de los dos nodos para que el tiempo estimado de ejecución para el siguiente grupo de 20 iteraciones sea aproximadamente igual. Este proceso se repite para todos los vecinos. Este proceso corresponde a una iteración del algoritmo GDE. Esto se repite un cierto número de veces que dependerá del diámetro de la topología.

Una vez obtenida la cantidad de carga que debe ser intercambiada entre los procesadores, es necesario migrar las moléculas que sean necesarias. Para migrar moléculas en la descomposición espacial de DM, se redefine el espacio asignado a cada procesador. Para ello, las moléculas de cada procesador son ordenadas de acuerdo a su distancia a los bordes actuales del subespacio de simulación. Una vez ordenadas las moléculas, se mueve el borde del espacio de forma tal que queden fuera del espacio físico del procesador y pasen a formar parte del espacio del procesador vecino.

4. EXPERIMENTOS Y RESULTADOS

El algoritmo fue probado sobre un patrón de prueba que consiste de 32000 moléculas simuladas en un paralelepípedo de tres dimensiones con condiciones periódicas de borde. Se basa en un modelo de Lennard-Jones con densidad reducida $\rho^* = 0.3$ y temperatura reducida $T^* = 0.8$. Dado que esta sustancia es homogénea y no presenta significativos niveles de desbalance, se cambió periódicamente la temperatura de la misma para forzar ciertos

niveles de desbalance. Los experimentos se realizaron sobre un cluster de cuatro nodos duales Pentium III de 800 MHz con 512MB de memoria RAM conectados mediante una red Myrinet.

En la figura 1, se muestran los tiempos de ejecución de cada 20 iteraciones de la simulación sin activación del balance de carga. A partir de la figura se observa que luego de la iteración 12000 las moléculas comienzan a concentrarse en el centro del paralelepípedo (nodos 1 y 2), esto hace que los procesadores 1 y 2 demoren más tiempo que los procesadores 0 y 3 en cada iteración de la simulación.

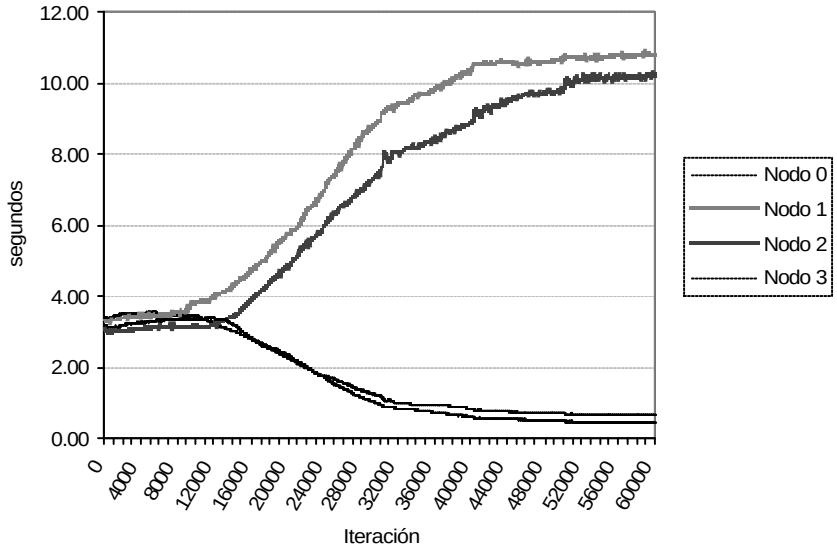


Figura 1. Tiempo de ejecución de cada 20 iteraciones (Tactual) sin Balance de Carga

En la figura 2, se presenta el comportamiento de la sustancia cada 20 iteraciones cuando se aplica el balance de carga. En este caso, el balance de carga se activa cuando el coeficiente de variación de T_{actual} es igual a 0.02. Cuando el COV es bajo, el balanceador se activa ante pequeños desbalances en el tiempo de ejecución de los procesadores. A partir de esta figura, podemos observar que los tiempos se reducen considerablemente en relación a la figura 1, y que los distintos procesadores que intervienen presentan un comportamiento bastante similar.

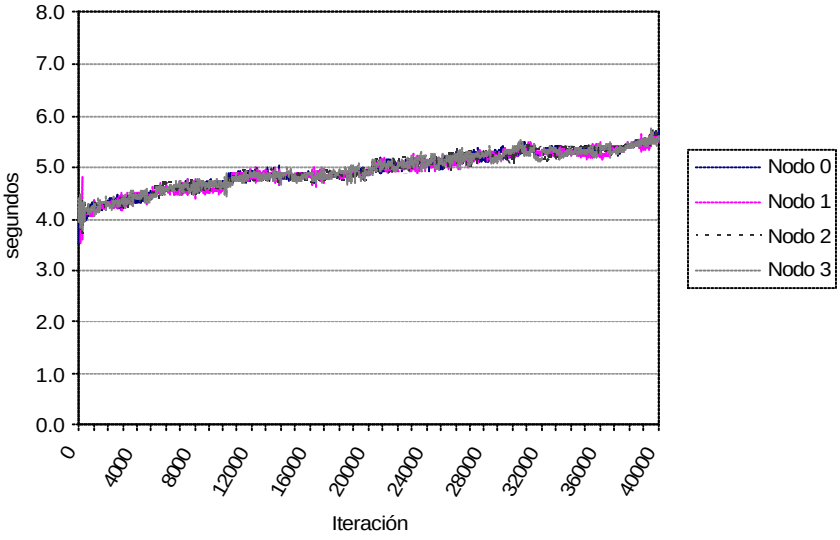


Figura 2. Tiempo de ejecución de cada 20 iteraciones (T_{actual}) con Balance de Carga activado con COV=0.02

En la figura 3, se muestra la variación en el número de moléculas presentes en cada procesador cuando se aplica el balance de carga. Podemos observar que aún cuando la variación entre los tiempos de ejecución de los procesadores es muy baja (se mantiene un balance con un COV=0.02 en tiempo), hay una variación alta en cuanto al número de moléculas (momentos en los que un proceso tiene alrededor de 7000 moléculas mientras otro tiene unas 9000).

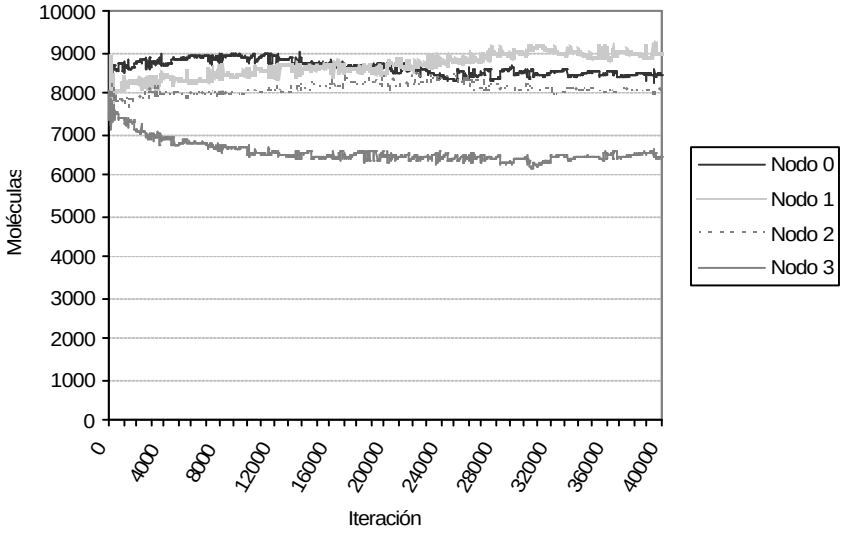


Figura 3. Número de moléculas por procesador con Balance de Carga activado con COV=0.02

La Figura 4 muestra para diferentes márgenes de tolerancia al desbalance, la diferencia en tiempo de ejecución cuando el balanceador utiliza las medidas de carga basadas en moléculas y en tiempo. La medida de carga basada en tiempo permite mejoras de hasta 6.5% sobre la basada en cantidad de moléculas. La escogencia del COV más apropiado depende de la cantidad de *overhead* que puede generar una migración de moléculas en un instante dado de la simulación y la ganancia estimada al lograr el balance del sistema [3].

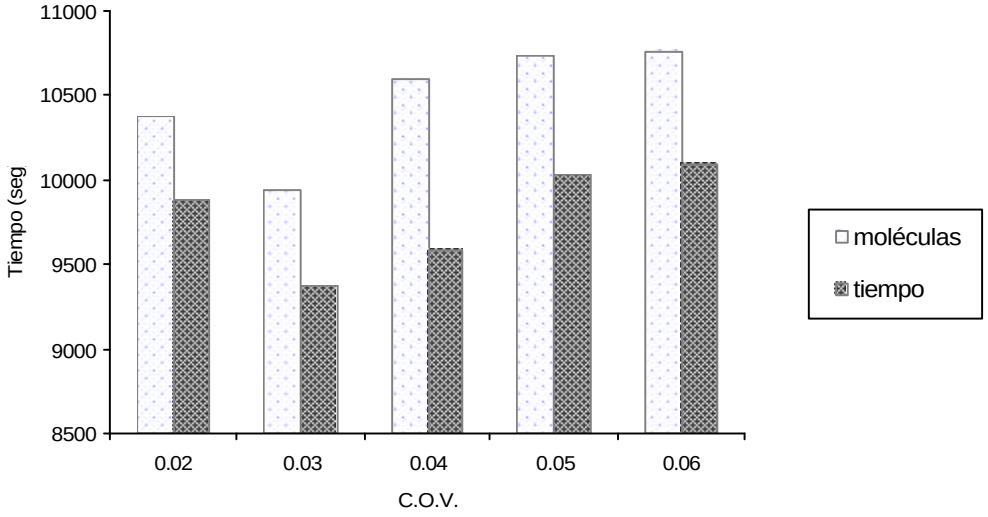


Figura 4 Comparación de tiempos de ejecución para medidas de carga basadas en moléculas y tiempo

La Tabla 1 muestra el tiempo total de ejecución y el porcentaje de disminución de tiempo obtenidos para distintos experimentos llevados a cabo variando el nivel de activación del balance. La disminución en el tiempo está entre 32 por ciento y 43 por ciento. Cuando el balance se activa con COV igual a 0.02, se está incurriendo en un mayor *overhead* debido a la constante migración de moléculas de un subespacio de simulación a otro. Cuando se activa con COV igual a 0.08, se permite un mayor desbalance por mayor cantidad de tiempo pero aún así se obtiene una disminución considerable en el tiempo total de ejcución.

	Balanceador utiliza medida de carga basada en tiempo				
	COV=0.02	COV=0.03	COV=0.04	COV=0.05	COV=0.08
Tiempo Total	10709.77	9752.77	9777.82	10174.38	11518.74
% disminución	37.47	43.05	42.91	40.59	32.74

Tabla 1. Tiempo total de ejecución y porcentaje de disminución de los tiempos

5. CONCLUSIONES

Aplicaciones dinámicas de cómputo intensivo como DM, requieren la migración de datos de un proceso a otro con miras a garantizar que todos los procesadores tengan una cantidad de trabajo equivalente. Cuando esta migración se logra de manera adecuada, se minimiza el tiempo de ejecución de la aplicación. En DM la cantidad de trabajo a realizar por cada procesador está directamente relacionada con el número de moléculas que tiene en su espacio de simulación. Por tanto, lograr que todos los procesadores mantengan aproximadamente el mismo número de moléculas, corrige las grandes discrepancias de tiempo que se originan de la natural reagrupación de las moléculas durante la simulación. Sin embargo, el tiempo total de ejecución no es óptimo. Nuestro trabajo logró optimizar el tiempo de ejecución utilizando una medida de carga relacionada con el tiempo de ejecución de un fragmento de ejecución en el procesador en particular.

Actualmente nuestros esfuerzos están dirigidos en la aplicación del balance de carga basado en tiempos sobre arquitecturas heterogéneas y en el ajuste de la regla de inicio de balance de carga.

Referencias

[1] Cortes, A., Ripoll, A., Senar, M., Pons P. and Luque, E. *On the Performance of Nearest Neighbor Load Balancing Algorithms in Parallel Systems.* Proceedings of the Seventh Euromicro Workshop on Parallel and Distributed Processing, Funchal, Portugal, Febrero 1999.

[2] Di Serio, A. and Ibánez, M.B. *Distributed Load Balancing for Molecular Dynamics Simulations*. Proceedings of the 16th Annual International Symposium on High Performance Computing Systems and Application, Moncton, New-Brunswick, Canada, June 2002. IEEE Computer Society, 284-289.

[3] Di Serio, A. and Ibánez, M.B. *Evaluation of a Nearest/Neighbor Load Balancing Strategy for Parallel Molecular Simulations in MPI Environment*. Recent Advances in Parallel Virtual Machine and Message Passing Interface. Volume 2474, (Octubre, 2002), pp. 226-233.

[4] Esselink, K., Smit, B. and Hilbers. *Efficient Parallel Implementation of Molecular Dynamics on a Toroidal Network: Multi-particle Potentials*. Journal of Computer Physics. Vol. 106, pp. 108-114, 1993.

[5] Finchman. *Parallel Computers and Molecular Simulation.* Molecular Simulation. Vol. 1, 1987.

[6] Haile, J.M. *Molecular Dynamics Simulation.* Wiley Inters-Science, 1992.

[7] Hegarty, D.F. and Kechadi, T. *Topology Preserving Dynamic Load balancing for Parallel Molecular Simulations.* Super Computing, 1997.

[8] Plimpton, S. *Fast Parallel Algorithms for Short-range Molecular Dynamics.* Journal of Computational Physics. Vol 117, pp. 1-19, 1995.

[9] Sato, N. and Jézéquel, J.M. *Implementing and Evaluating an Efficient Dynamic Load-Balancer for Distributed Molecular Dynamics Simulation.* Proceedings of the 2000 International Workshops on Parallel Processing. IEEE, 2000.

[10] Willebeek-LeMair, M. and Reeves, A. *Strategies for Dynamic Load Balancing on High Parallel Computers.* IEEE Transactions on Parallel and Distributed Systems, 9(3): 235-248, Marzo 1998.

[11] Xu, C. and Lau, F.C.M. *Load Balancing in Parallel Computers. Theory and Practice.* Kluwer Academic Publishers, 1997.

Arquitectura de Sistemas de Información basados en Componentes sobre la Plataforma J2EE

Daniel Perovich, Leonardo Rodríguez, Andrés Vignaga
Universidad de la República, Facultad de Ingeniería, Instituto de Computación
Montevideo, Uruguay, 11300
{perovich,lrodrigu,avignaga}@fing.edu.uy

Resumen

El desarrollo de sistemas basado en componentes puede ser atacado desde dos frentes, o más precisamente, niveles diferentes. Uno de ellos es el nivel de la tecnología que se empleará para la implementación del sistema, y el otro es un nivel previo más abstracto en el que el centro es la estructura lógica de la solución dejando de lado aspectos particulares de la tecnología. El enfoque de Model Driven Architecture incorpora esta separación distinguiendo modelos de sistemas que son independientes de la plataforma de desarrollo de los que son específicos para la misma. Alineado con este enfoque, este artículo propone una correspondencia entre la arquitectura lógica en capas de un sistema de información creado independiente de la tecnología aplicando una propuesta metodológica conocida, y las construcciones de la plataforma J2EE. Esta correspondencia o mapping permite definir transformaciones entre modelos independientes de la plataforma resultantes de la aplicación de la metodología mencionada que permiten un razonamiento abstracto de la solución, y modelos específicos de la plataforma que se encuentran alineados con las construcciones de la tecnología y que son implementables en forma directa.
Palabras claves: Arquitectura de software, Desarrollo basado en componentes, Sistemas de información, Java 2 Enterprise Edition, Enterprise Java Beans, Model Driven Architecture.

Abstract

Component-based development can be addressed from two different fronts, or more precisely, from two different levels. One of them regards the technology used for system implementation, and the other is a previous and more abstract level, where the focus is set to the logical structure of the solution and where technological issues are not considered. Model Driven Architecture promotes such separation by distinguishing platform independent models from platform specific models. In alignment with this approach, this article proposes a mapping between the tiered and platform independent architecture for information systems resulting from the application of a widely known methodological approach, and the available constructs in the J2EE platform. This mapping allows the definition of transformations between platform independent models, resulting from the referred methodology where it is possible to abstractly reason about the solution, and platform specific models which are aligned with technological constructs and are directly implemented.
Keywords: Software architecture, Component-based development, Information systems, Java 2 Enterprise Edition, Enterprise Java Beans, Model Driven Architecture.

1. INTRODUCCIÓN

El desarrollo basado en componentes (CDB) se encuentra en permanente evolución. En particular, dicha evolución se experimenta en dos mundos diferentes. Por un lado, puede encontrarse estudios del CDB a nivel conceptual y/o metodológico, en los que fijando la noción de componente se busca definir la arquitectura lógica de ciertos tipos de sistemas y los pasos para poblarla con componentes para un sistema particular de uno de esos tipos. Por otra parte, se encuentran las tecnologías que proporcionan plataformas para el desarrollo y ejecución de componentes (o sistemas basados en componentes), como por ejemplo Java 2 Enterprise Edition (J2EE) [13]. En este mundo, la noción de componente se encuentra definida como una construcción de implementación. A pesar de que las plataformas existentes presentan puntos en común, cada una impone finalmente su enfoque particular, el cual va evolucionando con cada nueva versión de la tecnología. Podría considerarse que el primer mundo es independiente del segundo, en el sentido que busca expresar soluciones en términos de nociones abstractas (independientes de la tecnología) de componentes, enfoque que se denomina "basado en modelos". Sin embargo, dichos modelos deberán ser implementados utilizando las construcciones que una cierta tecnología brinde, por lo tanto se hace necesaria una conexión entre ambos mundos.

Es posible concebir una solución completamente en el mundo tecnológico, pero esto trae consigo algunos inconvenientes. Por ejemplo, la solución queda atada a una tecnología particular, o más aún, a una cierta versión de la tecnología. El razonamiento de la solución se podría ver distorsionado por el "ruido" introducido por la terminología y enfoque de la tecnología. Además, requeriría que el arquitecto de software no solo contara con un conocimiento adecuado de la tecnología a aplicar, sino que además fuera experto en la misma. El enfoque basado en modelos permite razonar la solución en forma abstracta e independiente de la tecnología y constituye un enfoque interesante para el manejo de la complejidad que representa el desarrollo de un sistema basado en componentes. Propuestas como la de [3, 20] se enmarcan en este enfoque.

Existen en la actualidad esfuerzos por contemplar ambos mundos. El Object Management Group (OMG) con la iniciativa de Model Driven Architecture (MDA) [9], busca separar la lógica de un negocio de la tecnología de la plataforma subyacente. De esa manera, aplicaciones independientes de la plataforma construidas según el enfoque de MDA pueden ser realizadas en diferentes plataformas propietarias. El concepto de modelo es ciertamente central en MDA. En particular se distinguen dos tipos de modelos: el PIM (platform independant model o modelo independiente de la plataforma), y el PSM (platform specific model o modelo específico de la plataforma). En MDA se transforma un PIM generado para un sistema en un cierto PSM. Luego, el PSM puede ser implementado directamente en la plataforma específica.

En este trabajo nos alineamos con el enfoque de MDA descrito. Partiendo de un modelo de un sistema realizado en forma independiente de la tecnología y según una metodología particular (es decir, un PIM generado aplicando una cierta metodología), buscamos definir una correspondencia con las construcciones de una tecnología particular. Eso quiere decir que el objetivo es brindar los elementos necesarios para la generación de un modelo específico de la plataforma (es decir la transformación a un PSM), y no la generación de código directamente desde el PIM. Concretamente los modelos PIM sobre los que nos concentraremos son los generados aplicando el enfoque de [3] el cual está a su vez basado en [4]. En él, se explican los pasos necesarios para especificar la arquitectura de componentes de un sistema de información, organizado en una arquitectura de capas, e independientemente de la tecnología. La plataforma para la cual se estudiará la transformación es J2EE. Esta transformación fue marginalmente estudiada en el propio [3], es nuestro objetivo dar un tratamiento tanto más profundo como más amplio a este punto.

El resto del documento se organiza de la siguiente forma. La sección 2 repasa las ideas fundamentales del enfoque metodológico de [3], y presenta un PIM resultado de su aplicación. En la sección 3 se presenta una correspondencia entre los tipos de componentes presentes en un PIM y las construcciones de J2EE, brindando una base para la generación de PSMs para dicha plataforma. La sección 4 presenta el PSM generado a partir de PIM de la sección 2 atendiendo lo expresado en la sección 3. La sección 5 concluye.

2. ARQUITECTURA DE SISTEMAS DE INFORMACIÓN INDEPENDIENTES DE LA TECNOLOGÍA

Esta sección contiene un repaso del las principales ideas que se manejan en [3]. La arquitectura de un sistema de información se especifica en dos niveles de refinamiento. El primero, denominado *Arquitectura del Sistema* expresa el estilo de arquitectura a aplicar en el nivel más alto de abstracción. Como se dijera previamente, el estilo aplicado es el de capas. En la primera parte se definen concretamente las capas a utilizar, así como las principales responsabilidades de los componentes que residirán en cada una de ellas. En la segunda parte, en base a las responsabilidades mostradas, se estudian los elementos básicos de una especificación de *arquitectura lógica* de componentes, que constituyen los modelos independientes de la tecnología (PIM), así como un ejemplo de uno de ellos.

2.1 Arquitectura del Sistema

Los sistemas de información pueden modelarse según el estilo de arquitectura de *niveles de abstracción* o *capas* [11]. Este estilo se organiza jerárquicamente. Cada capa brinda servicios a la capa superior a ella y actúa como

cliente para la capa inferior. Los servicios brindados en las capas altas corresponden a un nivel alto de abstracción, y los servicios brindados por capas bajas corresponden a un nivel bajo de abstracción. A continuación se presenta la organización de un sistema de información indicando las responsabilidades de los componentes que residen en cada capa. Esta organización está dada de forma que las capas están presentadas en orden decreciente de abstracción, y está basada en la propuesta por [3] habiéndosele agregado la capa de *IU de Diálogos del Usuario*:

- Capa de Interfaz de Usuario: Se encarga de la presentación al usuario. Contiene formularios, páginas Web, etc.
- Capa de IU de Diálogos del Usuario: Maneja la lógica de la IU y coordina la presentación al usuario.
- Capa de Diálogos del Usuario: Maneja la lógica de diálogos y mantiene el estado del diálogo.
- Capa de Servicios del Sistema: Exporta operaciones que resuelven requerimientos del sistema. Está organizada en subsistemas. Contiene además adaptadores a sistemas externos.
- Capa de Servicios del Negocio: Los componentes corresponden a tipos estables del negocio. Administra la información persistente del negocio.
- Capa de Infraestructura: Provee servicios básicos como seguridad, poder transaccional, caching de datos, etc. Comprende en general a frameworks, APIs, manejadores de bases de datos, etc.

2.2 Arquitectura Lógica

La arquitectura del sistema presentada antes indica el estilo general que seguirá la aplicación. El siguiente paso es determinar el estilo que cada capa seguirá internamente, así como identificar que elementos la poblarán.
La organización interna utilizada en cada capa varía. La interfaz de usuario, si es orientada al Web, seguirá el estilo *cliente-servidor*. La metodología de desarrollo puede condicionar el estilo de arquitectura a elegir, ya que mediante la aplicación de la metodología cada capa será poblada con elementos que residirán en ella. El ejemplo de la Figura 1, tratado en [10], fue desarrollado aplicando el enfoque de [3, 20]. En consecuencia, el estilo de arquitectura utilizado en las capas Diálogos de Usuario, Servicios de Sistema y Servicios de Negocio es *componentes*.

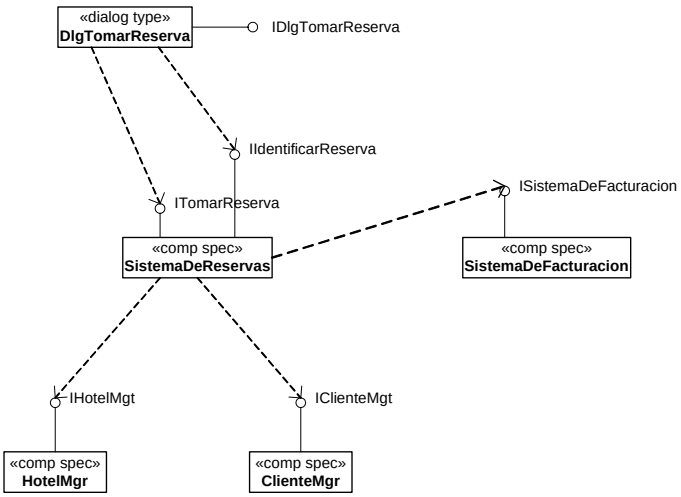


Figura 1 – Arquitectura lógica independiente de la tecnología

El estereotipo «comp spec» denota a una especificación de componente. Cada especificación ofrece una o más interfaces, las cuales son estereotipadas con «interface type». Una interface type tiene asociado un modelo de información sobre el cual se expresan los contratos de software para cada una de sus operaciones. Los diálogos de usuario no son considerados en [3]. Un diálogo de usuario, estereotipado con «dialog type», encapsula la lógica de un caso de uso. Por no tener un modelo de información asociado, la interfaz provista no necesita ser especificada mediante un interface type, una simple interfaz es suficiente para lo que se necesita especificar de un diálogo.

3. CORRESPONDENCIA CON LA PLATAFORMA J2EE

En esta sección se discuten los detalles de la correspondencia entre los elementos de cada una de las capas de la arquitectura del sistema y las construcciones de la plataforma J2EE. Con esta correspondencia es posible transformar un modelo tipo PIM, como el mostrado en la arquitectura lógica, en uno específico para J2EE tipo PSM. Al aplicarse EJB [14] a las capas que utilizan el estilo de componentes, para su modelado se aplicó el profile de UML para EJB [5]. Se atacarán una a una las capas indiadas en la arquitectura del sistema, basando los ejemplos en la arquitectura lógica ya presentada.

3.1 Interfaz de Usuario
Esta capa no fue incluida en la arquitectura lógica. Con el avance de las tecnologías de Internet, cada vez más empresas orientan sus aplicaciones al web, ya que cuentan con una Intranet. Teniendo esto en cuenta, realizaremos el mapeo de forma de utilizar tecnología web.
Esta capa se organiza generalmente con el estilo cliente-servidor. En el cliente corre un browser. Este despliega al usuario final páginas web basadas en HTML y HTML Forms. Mediante estos últimos se recopila información que será comunicada al servidor.
A nivel del servidor se cuenta con un servidor Web capaz de generar páginas web en forma dinámica. La tecnología Java Server Pages (JSP) [15] es el mecanismo usado para este fin. Las páginas JSP se construyen a partir de un lenguaje de tags y permite definir páginas dinámicas a través de templates. Estos templates incluyen tags HTML para la parte estática y tags especiales JSP para las partes en donde se debe incluir contenido dinámico. Para una interacción completa Actor-Sistema (i.e. un caso de uso) es necesario, en general, más de una página web. Se utilizará una página JSP por cada etapa de la interacción. Los forms serán dirigidos (mediante el correcto seteo del atributo `action`) a una URL atendida por un servlet [16] ubicado en la capa inferior. A su vez, este servlet es el responsable de proporcionarle a la página la información necesaria para generar el contenido dinámico. Para esto se utiliza el EJB pattern Data Transfer Object (DTO)1 [8]. Este define que la información debe estar empaquetada en clases serializables, para lo cual utilizaremos JavaBeans [12]. Este en general será el mecanismo utilizado para el intercambio de información entre todas las capas del sistema.

3.2 UI de Diálogos de Usuario
Esta capa aplica una variante del GRASP [7] facade controller. Todas las páginas dirigen sus respuestas a este controlador. Se utiliza un servlet para este fin2; el cual es el encargado de recibir la información del usuario (recibida mediante la solicitud http) e invocar al diálogo de usuario (use-case controller) correspondiente al caso de uso. El servlet a través de su objeto sesión tiene asociado un diálogo de usuario al cual redirigirá todas las solicitudes. En cada solicitud envía la información recibida (encapsulándola en un JavaBean), obtiene el resultado, y responde al usuario con una nueva página JSP, la cual se construye con los datos recibidos del diálogo.
Notar que de la forma en que se define este mapeo, es una aplicación del pattern Model-View-Controller [2] en donde las páginas JSP cumplen el rol de "View", el servlet cumple el rol de "Controller" y las capas que se presentan a continuación el rol de "Model". Otra alternativa al modelado de estas dos capas es utilizar el framework Struts [6], lo cual se propone como trabajo futuro.

3.3 Diálogos de Usuario
El diálogo de usuario encapsula la lógica del caso de uso. Así, implementa una máquina de estados (derivada a partir del documento de casos de uso expandido), la cual es utilizada para saber qué servicios y en qué orden requerir a la capa inferior, así como para determinar la próxima página a visualizar. El servlet definido para la capa anterior no implementa esta máquina de estados. No vamos a exigirle que conozca cual es la operación en el diálogo que corresponde invocar a partir de una solicitud. Para ello vamos a aplicar el design pattern Command. Definimos una interfaz `IDialog` cuya única operación es `execute(in datain:Data):Result`.
`Data` es la superclase de todos los datos que son intercambiados entre cada diálogo y la capa superior, y se define como JavaBean. Así mismo, `Result` es otro JavaBean que contiene una instancia de `Data`, que representa a los datos resultantes, y además un identificador del estado en que quedó el diálogo luego de terminar la ejecución de `execute`. El servlet (en la capa superior) utiliza el identificador del estado para elegir la siguiente página JSP, a la cual le provee la instancia de `Data` incluida en la instancia de `Result` recibida. Notar que este mapping puede definirse mediante un archivo de configuración (basado en XML por ejemplo), de forma de no necesitar cambiar el código del servlet si sufre cambios las páginas JSP o los diálogos de usuario. Un diálogo de usuario es implementado mediante un *stateful session bean (SSB)*, aplicando el design pattern State. El SSB es un tipo de EJB que permite mantener el estado conversacional con el cliente (en nuestro caso la máquina de estados) a lo largo del proceso.
En resumen, un escenario de acción entre las tres capas especificadas hasta el momento es el siguiente. El usuario llena los datos del formulario presentado en su browser. Al hacer submit la información viaja al servidor web y es recibida por el servlet. El mismo invoca la operación `execute` del diálogo con el cual fue instanciado pasándole el JavaBean (instancia de `Data`) que encapsula la información recibida del submit realizado por el usuario. El diálogo, en función de su estado y de la información recibida, solicita servicios a la capa Servicios del Sistema, obtiene un resultado, decide su siguiente estado, y retorna la instancia de `Result` que incluye la instancia de `Data` correspondiente al resultado y el identificador del nuevo estado. Así, el servlet dirige al cliente web a la página JSP correspondiente al estado recibido, construyéndola con la información recibida en la instancia de `Data`.

[1] O una de sus variantes.

[2] Los J2EE BluePrints recomiendan que los servlets sean utilizados en actividades con una importante componente de programación. Esto lo deja como el candidato ideal para cumplir el rol de controlador de procesos.

Es importante distinguir que granularidad de transaccionalidad es necesaria. Utilizando *container-managed transaction* (transaccionalidad manejada por el contenedor), solamente es posible demarcar transacciones a nivel de operaciones. Esto significa que si utilizamos transaccionalidad manejada por el contenedor para la interfaz del dialogo de usuario una transacción solo puede estar relacionada a un método de esta. En cambio, lo que se necesita es que el caso de uso entero este enmarcado dentro de la misma transacción. Para ello se debe utilizar *bean-managed transaction* (transaccionalidad manejada por el bean). En este caso, cuando se recibe la primera invocación a la operación `execute`, se crea una nueva transacción. Como se está utilizando un stateful session bean, la transacción será preservada entre una invocación y otra. La máquina de estados implementada en el bean podrá indicar cuando debe realizarse el commit, o hacer rollback en caso de que el caso de uso falle.
Se presenta a continuación la vista interna del stateful session bean `DlgTomarReserva`.

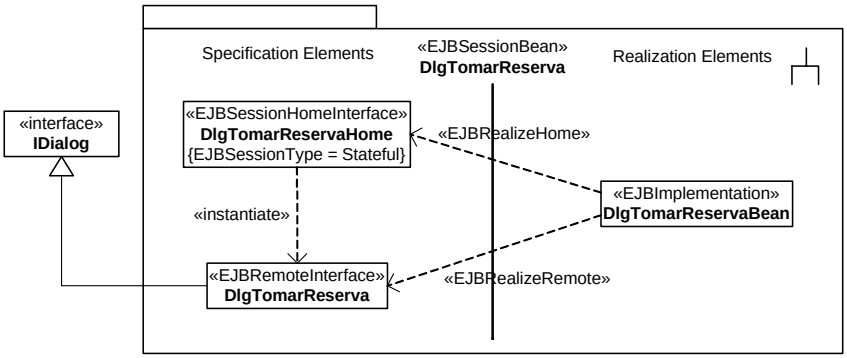


Figura 2 – Vista interna del session bean `DlgTomarReserva`

Notar que la Component Interface[3] del diálogo de usuario extiende la interfaz IDialog mencionada anteriormente.

3.4 Servicios de Sistema

Un proceso de negocios da lugar a un conjunto de casos de uso. Siguiendo la metodología, se crea una interfaz de sistema por cada caso de uso detectado. Estos casos de uso, originados a partir del mismo proceso, presentan una alta cohesión. Así, se crea un único componente de sistema que realiza todas las interfaces originadas por dicho proceso.
En el modelo de componentes EJB cada componente implementa una y solo una Component Interface. Para lograr que que un componente implemente más de una interfaz, es posible que la Component Interface sea sub-interfaz de todas las interfaces provistas por el componente. Determinar si la Component Interface es Remota o Local no es un tema menor4. En el común de los casos, en donde la capa de "Diálogos de Usuario" está ubicada en el mismo servidor de aplicaciones que la de "Servicios del Sistema", lo más performante es hacer que la Component Interface sea Local. Si por otro lado, el caso es que interesa distribuir estas capas sería necesario tener interfaces Remotas. Este mismo criterio se utilizará para determinar si las interfaces de las capas inferiores deben ser remotas o locales.
Para el caso de estudio se considerará que las capas de "Dialogo de Usuario", "Servicios de Sistema" y "Servicios de Negocio" están ubicadas en el mismo servidor de aplicaciones, por lo cual se utilizarán interfaces Locales.
El componente creado actúa de fachada del sistema para el proceso, agrupando todas las operaciones involucradas en él. Los componentes en esta capa cumplen el rol de *Facade Controller* siguiendo GRASP nuevamente.
La información de la sesión referente al diálogo entre el Actor y el Sistema es mantenida en la capa superior; los componentes a este nivel no necesitan mantener estado. Por ello se utilizará *stateless session beans*. Notar que la capa de servicios del sistema sigue los lineamientos del EJB pattern Session Facade [8].
Es importante notar que generalmente todas las operaciones a este nivel necesitan estar enmarcadas en una transacción. Esto se indica en forma declarativa marcando cada una de las operaciones con el atributo `TX_REQUIRED`.
Se presenta a continuación la vista interna del stateless session bean `SistemaDeReservas`.

[3] El modelo de componentes EJB define que un componente debe ser accedido a través de su Component Interface. Esta representa la especificación que el componente realiza. También define dos variantes, la Remote Interface que permite que el componente sea accedido remotamente y la Local Interface que permite que el componente sea accedido localmente.

[4] Notar que esto no sería un problema si la especificación EJB permitiera manejar transparentemente el hecho que el componente sea accedido remota o localmente. Esto claramente es una limitante de la especificación actual.

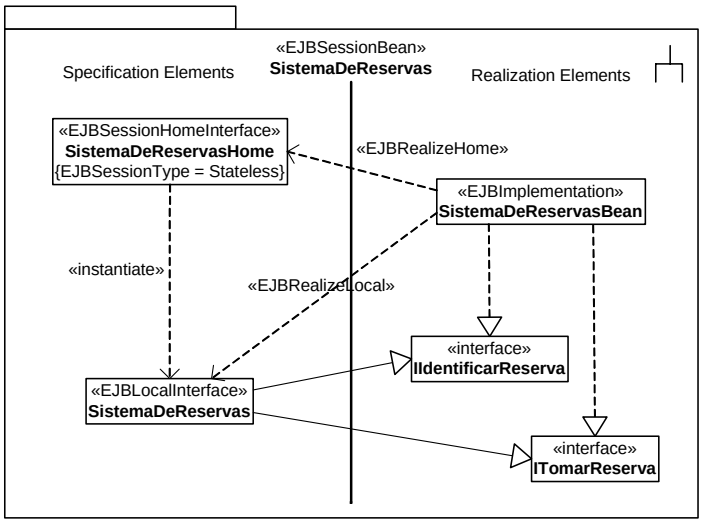


Figura 3 – Vista interna del session bean `SistemaDeReservas`

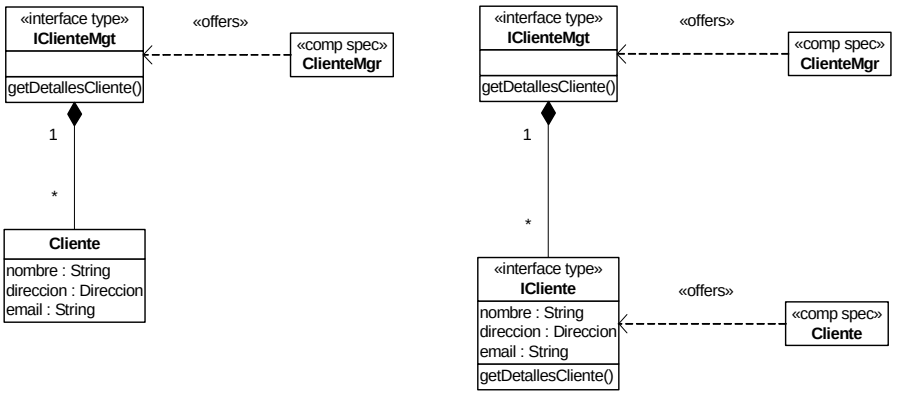
3.5 Servicios de Negocio

Los componentes en esta capa fueron detectados a partir de un Modelo de Tipos del Negocio. La metodología indica que se debe crear una interfaz de negocio responsable de manejar todas las instancias de cada tipo principal (conocido como *core type*) del modelo (e.g. `IClienteMgr`). Para acceder y procesar dichas instancias debe pasársele al manager los identificadores (e.g. `ClienteId`) de las instancias sobre las cuales trabajar. Por ejemplo, una operación `getDetallesCliente()` de `IClienteMgr` recibe el identificador del cliente y retorna los detalles del mismo.

Al implementar un manager en EJB se utilizará también un stateless session bean. Cada componente que necesite servicios del manager deberá solicitar una instancia del mismo. Este enfoque es simple, aplicable en muchos casos y da soporte a las interfaces tal cual fueron especificadas.

Una vez decidido que es un session bean, debe definirse en que forma el manager va a manipular la información que administra y se encuentra en una o varias fuentes de datos (*data sources*).

Un primer enfoque es que los session bean utilicen Java DataBase Connectivity (JDBC) [17] para manipular la información. Una clara desventaja de este enfoque es que el desarrollador debe implementar explícitamente la persistencia de su aplicación. Lo cual lleva a que no se logre una clara separación en lo que es la manipulación de los tipos de negocio y la persistencia de los mismos. Aparte puede ocasionar problemas conocidos como SQL-Spagueti. Para minimizar estos problemas se puede aplicar el J2EE pattern DataAccessObject [1] de forma de encapsular el acceso a las fuentes de datos. Otra alternativa interesante para manejar la persistencia es utilizar Java Data Object (JDO) [8, 19] como mecanismo de persistencia. El mismo provee un mapeo automático de un modelo de objetos en Java al modelo relacional. Aparte soporta características interesantes como transacciones y seguridad. Si bien esta tecnología no es parte de la plataforma J2EE puede ser integrada fácilmente, como se explica en [8]. Profundizar en JDO escapa al alcance del presente artículo. Ambos enfoques tienen en común que desde el punto de vista de los componentes, el session bean que actúa como manager es el responsable de manejar la persistencia de su información. La parte (a) de la siguiente figura esboza este enfoque.



(a) Persistencia manejada por el Manager (b) Persistencia manejada con Entity Beans

Figura 4 – Dos enfoques para implementar los servicios de negocio

Una alternativa a este enfoque es continuar la aplicación del modelo de componentes bajo el nivel del manager, considerando a las instancias de `Cliente` como instancias de componente (*component object*) propiamente. Con este enfoque se simplifica a su vez la especificación de alguna de las funcionalidades del manager ya que habrá menos necesidad de des-referenciar instancias. Las operaciones específicas a una instancia serán definidas en una interfaz que realice al interface type del Cliente, a saber `ICliente`. Podemos utilizar aquí un entity bean para implementar la interfaz `ICliente`, donde cada instancia del entity bean representa a un Cliente. Esto nos permite utilizar *container-managed persistente* para la recuperación y almacenamiento de las estructuras de un cliente. A su vez, a partir de EJB 2.0 [14], podemos utilizar EJB Query Language (EJB-QL) para consultar la información independizándonos así del lenguaje de consulta de la fuente de datos. En la parte (b) de la Figura 4 – se esboza este enfoque.
Este enfoque permite aprovechar las bondades del contenedor de EJB a nivel de las entidades del negocio (seguridad, transacciones, pooling, cache,, fail-over, clustering5, entre otras). En particular promueve la portabilidad de los componentes, la cual se vería afectada en el caso de utilizar JDO (por no ser una tecnología J2EE) o JDBC (si cambiamos el motor de base de datos subyacente).Pero por otro lado, para determinadas operaciones puede ser muy ineficiente, por ejemplo, las operaciones que afectan varios entity beans. Está fuera del alcance de este trabajo una completa comparación de ambos enfoques. Por más información ver [8].
El enfoque que se utilizará para modelar el resto del caso de estudio será el basado en entity beans. La vista interna del session bean `ClienteMgr` se presenta a continuación.

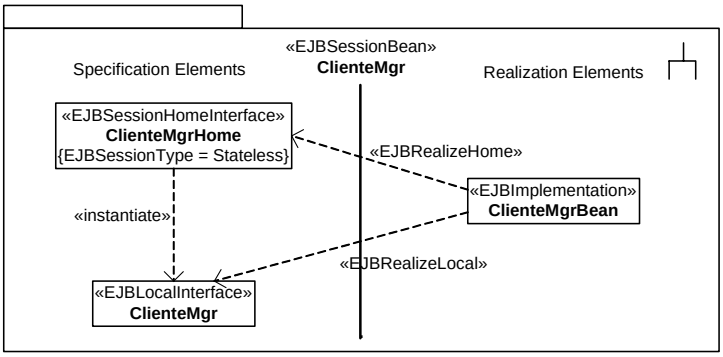


Figura 5 – Vista interna del session bean `ClienteMgr`

La vista interna del entity bean Cliente se presenta a continuación.

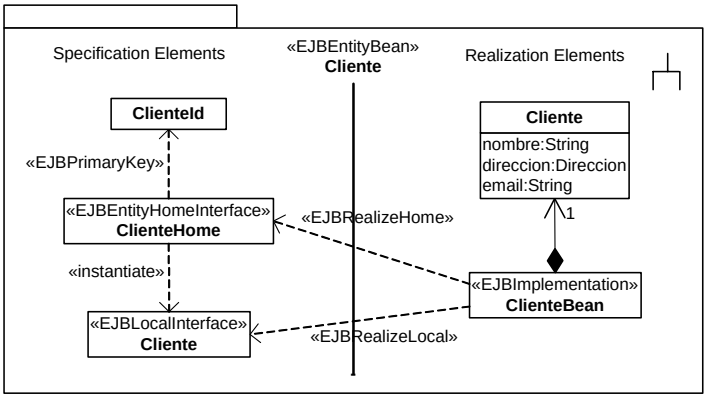


Figura 6 – Vista interna del entity bean `Cliente`

Cabe notar que tanto los entity beans como los session beans de esta capa exportan sus servicios mediante interfaces locales. En el caso particular de los entity beans, se recomienda que siempre sean accedidos a través de sus interfaces locales, lo cual implica que la capa de servicios de negocio estén en el mismo contenedor.

[5] Cabe notar que Fail-Over y Clustering no son funcionalidades estándares de un contenedor, pero la mayoría las implementan.

3.6 Integración con Sistemas Existentes

La integración con sistemas existentes puede realizarse utilizando las APIs provistas por el sistema existente o software para la integración de aplicaciones (EAI – Enterprise Application Integration).

El caso de estudio en consideración utiliza el sistema existente de Facturación en uso en la empresa. Será necesario desarrollar un adaptador (aplicación del design pattern Adapter) que provea un puente entre el entorno de componentes y el sistema externo. Siguiendo con el mapeo a la tecnología EJB, se utilizará J2EE Connector Architecture (JCA) [18] para acceder vía Common Client Interface (CCI) a las funcionalidades provistas por el sistema externo. Utilizando JCA, y asumiendo que se dispone del adaptador XA-Resource para el sistema existente, puede incluírselo en las transacciones.

Se incluirá entonces un session bean que dará soporte a la interfaz necesaria del sistema existente. Este bean redireccionará (probablemente necesite adaptar datos que entran y salen) las solicitudes al sistema externo. De esta forma centralizamos en un único componente el uso de JCA para acceder al sistema externo, preservando así las cualidades deseadas de un sistema basado en componentes.

La vista interna del session bean `SistemaDeFacturacion` se presenta a continuación.

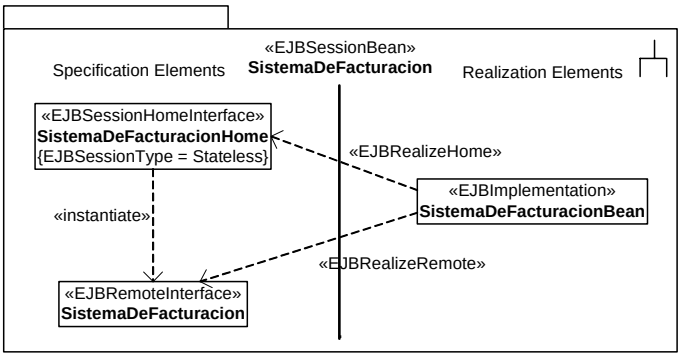


Figura 7 – Vista interna del session bean `SistemaDeFacturacion`

4. EJEMPLO DE MODELO ESPECÍFICO PARA LA PLATAFORMA J2EE

En esta sección se presenta el modelo tipo PSM correspondiente al modelo tipo PIM presentado en la Figura 1, el cual se muestra en la Figura 8. Esta vista unifica las distintas correspondencias estudiadas en la sección 3, refiriéndose solamente a aquellas capas pobladas con componentes, i.e. Diálogos, Sistema y Negocio.

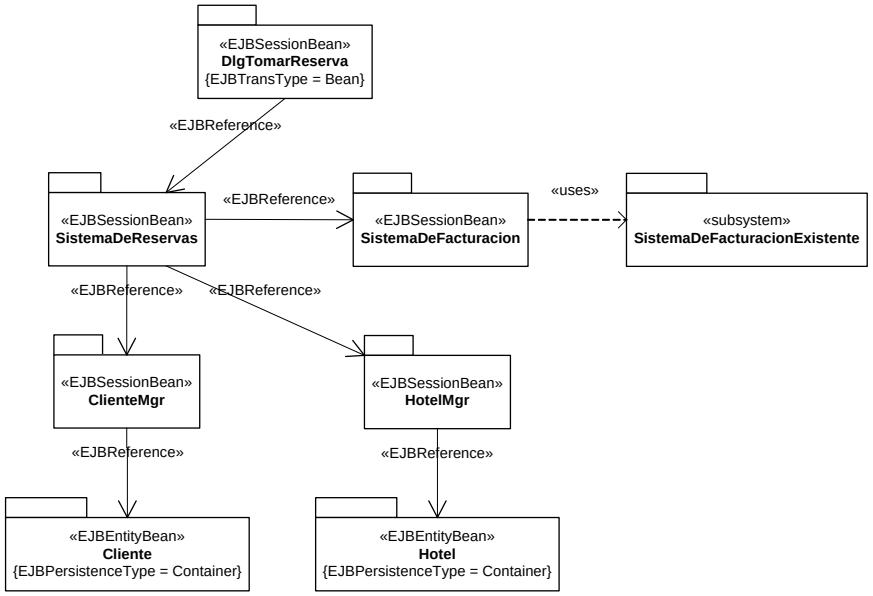


Figura 8 – Arquitectura lógica específica para J2EE

En el caso de estudio abordado en este trabajo, puede observarse que los modelos de cada figura presentan una estructura similar. El hecho de que en la Figura 8 se cuente con dos beans correspondientes a un componente de negocio (e.g. la especificación de componente ClienteMgr se corresponde con el session bean ClienteMgr y el entity bean Cliente) refiere a haber optado por utilizar entity beans; de la aplicación de un enfoque basado en JDBC o JDO no se obtendría el mismo modelo.

5. CONCLUSIONES

La arquitectura de los sistemas de información suele organizarse según el estilo de capas. La cantidad de capas y la responsabilidad de cada una dependen del enfoque metodológico aplicado para el desarrollo del sistema. En [3] se define una arquitectura para dichos sistemas y se presenta una propuesta orientada a la especificación independiente de la tecnología de implementación de componentes como los habitantes de algunas de las capas. Una arquitectura de especificación para un sistema particular corresponde a un PIM en el enfoque de MDA. En este artículo se refinó la arquitectura propuesta para los sistemas de información añadiendo una nueva capa, la capa de *IU de Diálogos del Usuario*, y se definió una correspondencia entre los elementos de cada una de las capas de la arquitectura del sistema y las construcciones de la plataforma J2EE. Esta correspondencia permite realizar una transformación a una arquitectura de especificación. El resultado de dicha transformación se expresa en términos específicos de la plataforma, y corresponde a un PSM en el enfoque de MDA. La separación entre PIMs y PSMs, y la definición de transformaciones de unos a otros constituyen la esencia del enfoque de MDA. Este trabajo presenta un estudio que hace posible una de esas transformaciones.

Finalmente, resulta de interés completar el ejemplo utilizado en el artículo, generando la arquitectura lógica específica para J2EE del caso de estudio de [10], con el objetivo de realizar una implementación completa que permita refinar la correspondencia presentada. También resulta de interés modelar las dos capas más superiores utilizando el framework Struts, así como el estudio de correspondencias análogas a otras tecnologías, como ser la plataforma .NET.

Referencias

[1] D. Alur, J. Crupi, D. Malks. *Core J2EE Patterns: Best Practices and Design Strategies (1st Edition)*. Prentice-Hall, 2001.

[2] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, M. Stal. *Pattern-Oriented Software Architecture, Volume 1: A System of Patterns*. John Wiley & Sons, 1996.

[3] J. Cheesman, J. Daniels. *UML Components: A Simple Process for Specifying Component-Based Software*. Addison-Wesley, 2001.

[4] D.F. D'Souza, A.C. Wills. *Objects, Components and Framekorks with UML: The Catalysis Approach*. Addison-Wesley, 1998.

[5] J. Greenfield. *UML Profile for EJB*. Rational Software Corporation, Public Draft. Internet: `http://www.jeckle.de/files/UMLProfileForEJBPublicDraft.pdf`. 2001.

[6] T. Husted, C. Dumoulin, G. Franciscus, D. Winterfeldt, C. R. McClanahan. *Struts in Action: Building Web Applications with the Leading Java Framework*. Manning Publications Company, 2002.

[7] C. Larman. *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process (2nd Edition)*. Prentice-Hall, 2001.

[8] F. Marinescu. *EJB Design Patterns. Advanced Patterns, Processes and Idioms (1st Edition)*. John Wiley & Sons, 2002.

[9] OMG. *Overview and Guide to OMG's architecture, MDA Guide Versión 1.0*. Object Management Group. Internet: `http://www.omg.org/docs/omg/03-05-01.pdf`. 2001.

[10] D. Perovich, A. Vignaga. *SAD del Subsistema de Reservas del Sistema de Gestión Hotelera*. Reporte Técnico RT03-15, InCo Pedeciba, Montevideo, Uruguay, 2003.

[11] M. Shaw, D. Garlan. *Software Architecture: Perspectives on an Emerging Discipline*. Prentice-Hall, 1996.

[12] Sun Microsystems. *JavaBeans Specification*. Versión 1.0.1. Internet: `http://java.sun.com/products/javabeans/docs/spec.html`. 1997.

[13] Sun Microsystems. *Java 2 Enterprise Edition Platform Specification*. Versión 1.3. Internet: `http://java.sun.com/j2ee/1.3/download.html`. 2001.

[14] Sun Microsystems. *Enterprise JavaBeans Specification*. Versión 2.0. Internet: `http://java.sun.com/products/ejb/docs.html`. 2002.

[15] Sun Microsystems. *JavaServer Pages Specification*. Versión 1.2. Internet: `http://java.sun.com/products/jsp/download/index.html`. 2001.

[16] Sun Microsystems. *Java Servlet Specification*. Versión 2.3. Internet: `http://java.sun.com/products/servlet/download.html`. 2001.

[17] Sun Microsystems. *Java DataBase Connectivity Specification*. Versión 3.0. Internet: `http://java.sun.com/products/jdbc/download.html`. 2001.

[18] Sun Microsystems. *J2EE Connector Architecture*. Versión 1.5. Internet: `http://java.sun.com/j2ee/connector/download.html`. 2002.

[19] Sun Micrsystems. *Java Data Objects Specification*. Versión 1.0. Internet: `http://jcp.org/aboutJava/communityprocess/final/jsr012/index2.html`. 2003.

[20] A. Vignaga, D. Perovich. *Enfoque Metodológico para el Desarrollo Basado en Componentes*. Reporte Técnico RT03-14, InCo Pedeciba, Montevideo, Uruguay, 2003.

Detección de Microcalcificaciones en Imágenes de Mamografías Usando Diferencia de Filtros Gaussianos Optimizados

Samuel A. Oporto Díaz
Universidad Nacional de Ingeniería
Facultad de Ingeniería Industrial y de Sistemas
Av. Túpac Amaru 210 Rímac, Lima - Perú
Teléfono: +51(1) 481-1424, +51(1) 460-6246, Fax: +51(1) 332-2696
soporto@aurigacorp.com.pe
al977969@mail.mty.itesm.mx

Abstract

Since the microcalcifications are primary indicators of presence of cancer of breast, its detection is important to prevent and treat the disease. This paper proposes a method for detection of breast microcalcifications in X-ray mammograms using Difference of Gaussian filters (DoG), the microcalcifications are small objects, and they appear as small and bright regions with irregular shape in the breast. Their diversity in their shape, their orientation, their size and localization in a dense mammogram are the cause of the major difficulty for their classification. The DoG filter allows improving the contrast between the regions of interest and the background regions so that the local contrast is evident. In one first stage apply the DoG filter to extract the potential regions and in second stage, these regions are classified using the following features: area, mean gray level, mean background gray level, relative contrast and compactness, the experimental results indicate that the potential regions are right.

Keywords: Microcalcification, DoG Filter, Classification, Mammogram, Feature Extraction.

Resumen

Dado que las microcalcificaciones son indicadores primarios de presencia de cáncer de mama, la detección de ellos es importante para prevenir y tratar la enfermedad. En este documento propongo un método para la detección de microcalcificaciones en imágenes de mamografías usando diferencia de filtros gaussianos (DoG), las microcalcificaciones son pequeños puntos densos rodeados de tejido normal, que aparecen brillantes en la imagen, la detección es particularmente difícil si el tejido circundante también es denso. El filtro DoG permite mejorar el contraste entre las regiones de interés y las regiones del fondo buscando que los máximos locales sean evidentes. En una primera etapa aplico el filtro DoG para extraer las regiones potenciales y en una segunda etapa estas regiones son clasificadas usando características tales como área, promedio de gris, promedio de gris del fondo, contraste relativo y compacidad, los resultados experimentales indican que las regiones potenciales identificadas en la segunda etapa cubren efectivamente las microcalcificaciones detectadas previamente por el especialista.

Palabras claves: Microcalcificación, Filtro DoG, Clasificación, Mamografía, Extracción de Características.

1. INTRODUCTION

El cáncer de mama es una de las mayores causas de mortalidad en mujeres tiene un incidencia de 25 por cada 100,000 personas, el 99 % de los afectados son mujeres, el 30 % de los quistes mamarios son malignos. En ausencia de una efectiva prevención, el diagnóstico precoz es un importante medio para reducir la mortalidad. La tasa de mortalidad va de 10 % en el primer año al 80 % en 5 años luego del diagnóstico (Gordo [16]).

Las mamografías están entre las principales técnicas usadas para el diagnóstico de cáncer de mama (Gordo [16], Almenteros [5], Calero [8]). Las hallazgos que pueden ser observados en una mamografía son: masas, calcificaciones, áreas de densidades asimétricas, distorsiones arquitecturales, conductos lactíferos prominentes o retracción del pezón (De Paredes [10]), este trabajo se enfoca en la detección de microcalcificaciones.

Las microcalcificaciones son pequeñas acumulaciones de calcio de de 0.1 mm a 2 mm de ancho, ellos son indicadores favorables de presencia de cáncer de mama y son usadas frecuentemente para el diagnóstico del carcinoma intraductal o carcinoma ductal in situ, tienen probada capacidad para detectar estadios

tempranos de la enfermedad. Entre el 30 % al 50 % del cáncer de mama en el mundo es diagnosticado debido a las microcalcificaciones (UCHC [28]).

La sensibilidad de las mamografías (la fracción de cáncer de mama que es detectada mediante una mamografía) es del 85 % al 95 %, pero su valor predictivo comparado con la biopsia (la fracción de lesiones biopsiadas que probaron ser malignas) está en el rango de 15 % al 30 %. Del 2 % al 22 % de los resultados positivos obtenidos mediante una mamografía a la primera vez, requirieron nuevas evaluaciones para confirmar los resultados y del 12 % al 78 % requirieron una biopsia (Mushlin [21], Shen [25]). Las causas de esta baja sensibilidad (Ganott [15]) se deben al bajo contraste entre el tejido canceroso comparado con el tejido parenquimal normal, a su pequeño tamaño, a su variada morfología y a posibles deficiencias en el proceso de digitalización de la imagen

El doble diagnóstico ha mostrado que mejora la sensibilidad en a los más 15 % (Ciatto [7], Elmore [14]), pero como cada mamografía debe de ser revisado por dos radiólogos, este procedimiento es ineficiente dado que se reduce la productividad individual del especialista, una alternativa viable es que el otro radiólogo sea un computador actuando como la segunda opinión, así los resultados obtenidos por el computador pueden ser confirmados o rechazados (Anttinen [2], Thurfjell [27]).

Diferentes técnicas de procesamiento digital de imágenes han sido usadas para resolver el problema Norhayati [18], Bazzani [1], Bocchi [3], Campanini [4], Cheng [6], Yu [29], El-naqa [13].

En este trabajo intento demostrar que una técnica de detección automática de microcalcificaciones incrementa la probabilidad de encontrar casos de carcinomas malignos, al desempañar el sistema una segunda opinión sobre el mismo caso. Con esta intención propongo un métodos para detectar microcalcificaciones (ver figura 1). El método está conformado por cuatro etapas: primero, el pre-procesamiento, permite eliminar aquellos elementos en la imagen que puede distorsionar el proceso de identificación de las microcalcificaciones, segundo, la detección de potenciales microcalcificaciones (señales), para este propósito se usa el filtro DoG, tercero, la extracción de características, permite extraer un conjunto de características desde cada señal y cuarto la etapa de clasificación, permite identificar si una señal obtenida corresponde o no a una microcalcificación individual, se usa una red neuronal para este propósito, obteniéndose finalmente las regiones con alta probabilidad de ser microcalcificaciones.

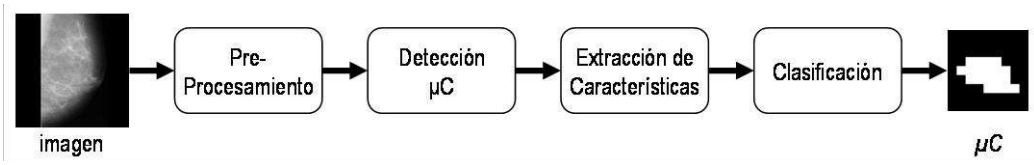


Figura 1: Esquema general del trabajo

Las imágenes de mamografías para este proyecto fueron tomadas de The Mammographic Image Analysis Society (MIAS [20]), la base de datos contiene 322 imágenes de las cuales 25 contienen microcalcificaciones, 13 son malignas y 12 son benignas, todas las imágenes son de 1024x1024 pixeles digitalizadas a 8 bits. Varios trabajos relacionados usaron esta base de datos: (Egan [12], Hayken [17], Karssemeijer [19], Rangarayyan [23]).

El problema que se pretende abordar en este trabajo de investigación es: Dado una base de datos de imágenes de mamografías, conteniendo algunas de ellas microcalcificaciones, cómo procesar las imágenes de tal forma que el conjunto de características a extraer maximice la eficacia en detectar las microcalcificaciones malignas. Para estimar el indicador de la eficacia del procedimiento propuesto, se aplica el estimador por validación cruzada de K conjuntos (Efron [11], Stone [26]), usado frecuentemente para probar esquemas de clasificación.

El resto del documento está organizado de la siguiente manera: en la sección 2 de este documento detallaremos la etapa de pre-procesamiento de la imagen, en la sección 3 exponemos el procedimiento usado para detectar, segmentar y selección de regiones consideradas como potenciales microcalcificaciones (señales), en la sección 4 veremos el procedimiento para extraer características desde cada señal identificada. sección 5 veremos el procedimiento de clasificación, en la sección 6 detallamos los resultados experimentales y en la sección 7 presentamos las conclusiones y recomendaciones del proyecto.

2. PRE-PROCESAMIENTO

Esta etapa tiene como finalidad eliminar aquellos elementos en la imagen que pueden distorsionar el proceso de identificar microcalcificaciones así como reducir el área de trabajo sólo a la región de la mama, para ello se aplica dos procedimientos: el filtro mediana y el corte automático, al procedimiento ingresa la

imagen original y se obtiene como resultados la imagen pre-procesada y una imagen espejo binarizada. Tiene tres parámetros: tamaño del filtro, tamaño de la ventana y umbral mínimo.

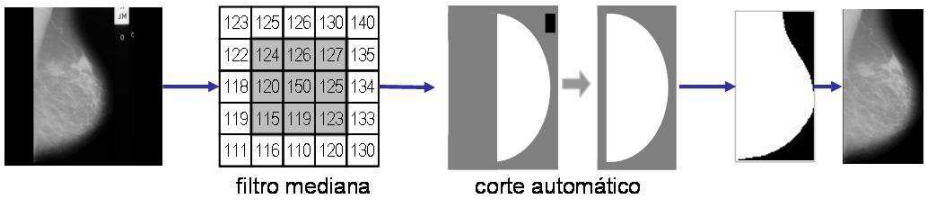


Figura 2: Diagrama de flujo. Etapa de Pre-Procesamiento

2.1. Filtro Mediana

El filtro mediana es un filtro no lineal, usado para eliminar el ruido de alta frecuencia sin eliminar las características significativas de la imagen, se usa una máscara de 3x3, la cual es centrada en cada pixel de la imagen reemplazando el pixel central por la mediana de los nueve pixeles que cubre la máscara, el tamaño de la ventana permite que se mantenga las características propias de la imagen y a la vez se eliminen las altas frecuencias. El filtro es superior a un filtro promediador, también usado para eliminar las altas frecuencias.

2.2. Corte Automático

La imagen es recortada para incluir sólo la región de interés, enfocando el proceso a la región que contiene a la mama y por lo tanto reduciendo el tiempo total de procesamiento, en el 90 % de las imágenes de la base de datos del MIAS la mama ocupa menos del 49 %.

El procedimiento se inicia generando un espejo de la imagen filtrada I, como si fuese una imagen binaria B, donde cada pixel representa una ventana de 16x16 en la imagen I, las dimensiones de esta imagen espejo son: $[filas\_imagen/lado\_ventana\ ,\ columnas\_imagen/lado\_ventana]$

Luego, se calcula el promedio de niveles de gris de cada ventana y si este es mayor que cierto $umbral$·se coloca 1 en B en caso contrario 0, el umbral se ha elegido mediante el desarrollo de experimentos ($umbral = 10$).

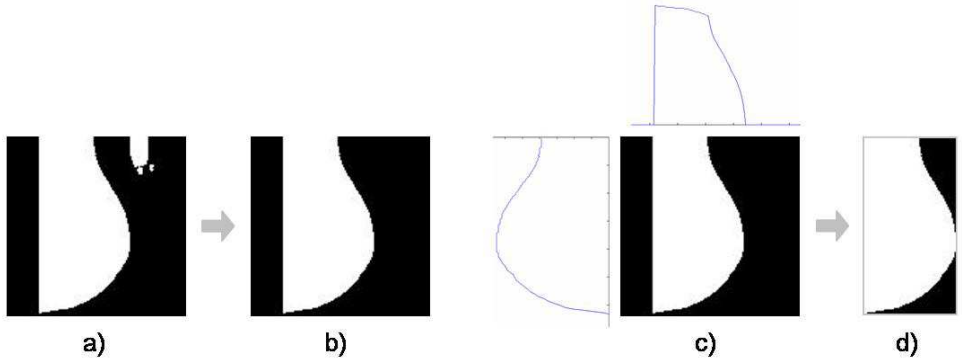


Figura 3: Resultados del corte automático. a). Imagen binaria (espejo), b). Imagen binaria sin regiones aisladas, c). Determinación de los extremos para ejecutar los cortes y d). Imagen binaria luego del corte horizontal y vertical

Para eliminar los elementos aislados en B, se calcula su centro de masa, bajo el supuesto que la mama contiene a su centro de masa (la mama es un objeto convexo). A partir del centro de masa se busca todos los pixeles conectados en vecindad V_4, Este último procedimiento nos permite extraer solo la región conectada que contiene a la mama B' (ver la figura 3 b).

Para regenerar la imagen de la mama I', se recorre la imagen binaria B', si el pixel visitado es 1, se copia la ventana correspondiente de I a I', el resto de regiones quedan en cero.

Finalmente para ejecutar el corte vertical se procede de la siguiente manera, se acumula verticalmente todos los pixels en B', cada posición del vector indica el número de pixeles que existe en la columna, luego se hace una búsqueda desde los extremos y se extrae la posición donde se ubica el primer pixel con valor diferente de cero tanto a la izquierda como a la derecha (figura 3 c). A continuación se ejecuta el corte

considerando las regiones entre estas dos posiciones, las misma operación se sigue para el corte horizontal. Este ultimo procedimiento se aplica tanto a la imagen binaria como a la imagen original.

3. DETECCIÓN DE MICROCALCIFICACIONES

Esta etapa tiene como propósito detectar las potenciales calcificaciones en la imagen y extraerlas en pequeñas ventanas de 9x9 pixeles. Dos métodos de selección son usados para descartar aquellas que pueden corresponder a ruido o a máximos locales no relacionados con calcificaciones.

El diagrama de flujo usado indicando, las entradas, salidas y parámetros necesarios se presenta en la figura 4, en total se aplica siete procedimientos: diferencia de filtros gaussianos, binarización global, etiquetado de regiones, selección de puntos por área mínima, segmentación de la imagen, selección de puntos por promedio de gris mínimo y binarización local.

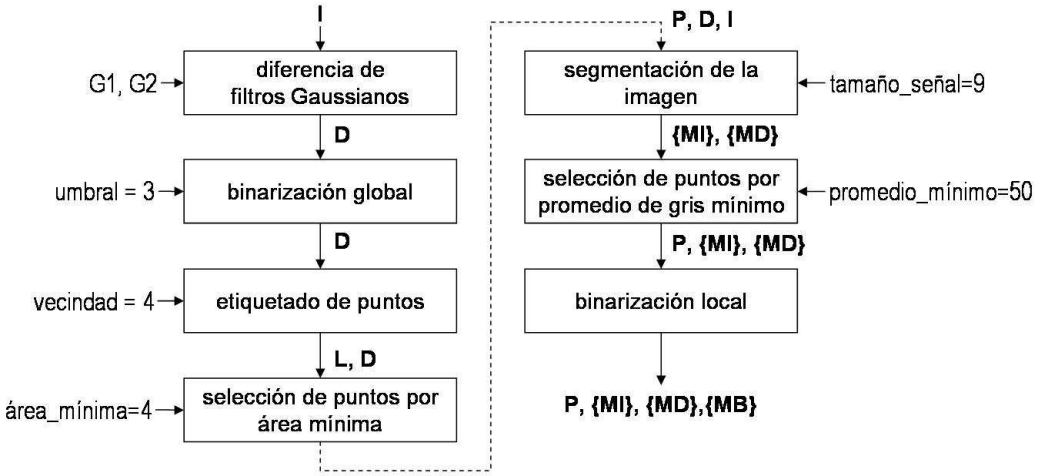


Figura 4: Diagrama de flujo. Etapa de detección de potenciales microcalcificaciones

En esta etapa los *puntos*, o regiones de alto contraste local todavía no pueden ser considerados como *microcalcificaciones*, deben de pasar por algunos procesos antes de adquirir tal categoría. En la figura 5 presentamos los diferentes nombres que adquieren en su evolución:

1. *Punto*. Son las las regiones brillantes detectadas en la imagen luego aplicar el filtro DoG y binarizar globalmente la imagen.
2. *Señal*. Son generados desde aquellos *puntos* que pasaron los dos procesos de selección (área mínima y gris mínimo) y luego de la binarización local.
3. *Calcificación*. Son aquellas *señales* que obtienen una ponderación positiva luego de pasar por un clasificador previamente entrenado.

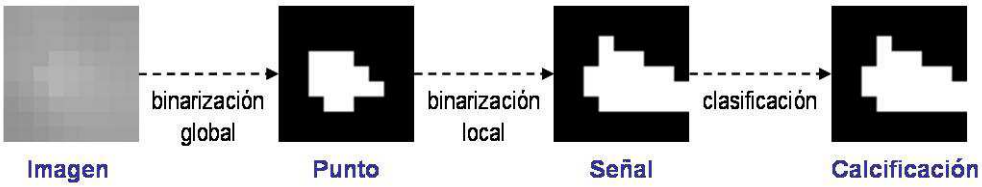


Figura 5: Nombres que adquieren los objetos identificados en la imagen

3.1. Diferencia de filtros gaussianos

La distribución gaussiana en 2-D tiene la siguiente fórmula: $G(x,y) = ke^{(x^2+y^2)/2\sigma^2}$, donde k es la altura de la función y σ es la desviación estándar.

El filtro DoG es un filtro pasabandas, en el dominio del espacio, construido a partir de dos filtros Gaussianos simples. Estos dos filtros deben tener varianzas diferentes (ver figura 7). Al sustraer las dos imágenes obtenidas de aplicar cada filtro por separado, se obtiene una imagen que contiene un rango de frecuencia que interesa detectar, de la siguiente manera: $DoG(x,y) = k_1 e^{(x^2+y^2)/2\sigma_1^2} - k_2 e^{(x^2+y^2)/2\sigma_2^2}$

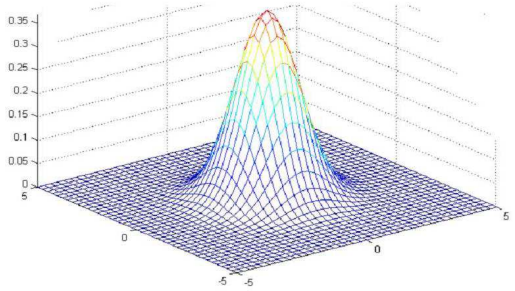


Figura 6: Ejemplo de una función de distribución gaussiana en 2-D

En este trabajo usamos dos filtros con desviaciones estándar: $\sigma_1 \approx$,0,7618 para la mascara de 5x5 (ver 7.a) y $\sigma_2 \approx$,0,8226 para la máscara de 7x7 (ver 7.b), dado que el filtro gaussiano es lineal se puede puede calcular la diferencia de las dos funciones gaussianas. Donde $k_1 = 9/53 \approx 0{,}1698$ y $k_2 = 16/140 \approx 0{,}1143$.

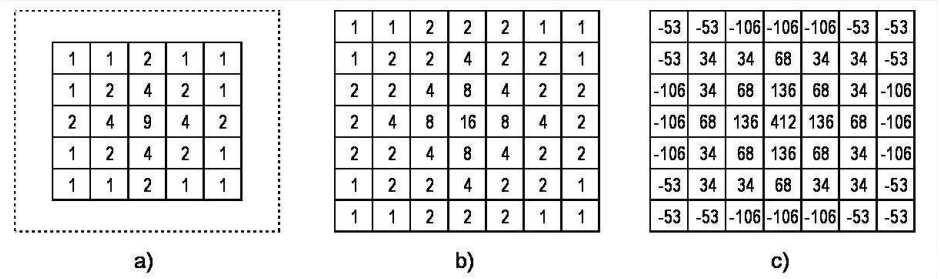


Figura 7: Mascaras Gaussianas usadas con el filtro DoG. a). Mascara 5x5 con $\sigma_1 \approx 0{,}7618$, b). Mascara 7x7 con $\sigma_2 \approx 0{,}8226$ y c). Mascara 7x7 del filtro DoG

3.2. Binarización global

Luego de restar las dos imágenes convolucionadas, procedemos a binarizarla, el objetivo del proceso de binarización es obtener una imagen en blanco y negro, ubicando un umbral adecuado (T) tal que aquellos pixeles menores que T correspondan a 0 y aquellos que tomen valores mayores o iguales que T correspondan a 1, donde R es la imagen obtenida luego de restar las dos imágenes convolucionadas y P_i es un pixel en la imagen.

$$D(P_i) = \begin{cases} 0 & \text{si } R(P_i) < T \\ 1 & \text{en otro caso} \end{cases} \quad (1)$$

El *umbral* de binarización para este problema fue asignado manualmente, dependiendo de los ensayos realizados ($umbral = 3$).

El histograma típico de la imagen generada por la aplicación del filtro DoG tiene un rango de valores entre $[-15, 30]$, existe fuerte acumulación de alrededor de los niveles 0, 1 y 2, por otro lado a medida que el nivel crece la frecuencia decrece, los niveles altos correspondan a puntos brillantes en la imagen.

En la figura 8, presentamos los resultados de aplicar diferentes valores para el *umbral*, así para $umbral > 5$, sólo se puede separar las regiones muy brillantes reduciendo la sensibilidad del proceso, se aplicó dos filtro gaussianos ($DoG(5x5) - DoG(7x7)$), la relación $\sigma_2/\sigma_1 = 0{,}8226/0{,}7618 = 1{,}08$.

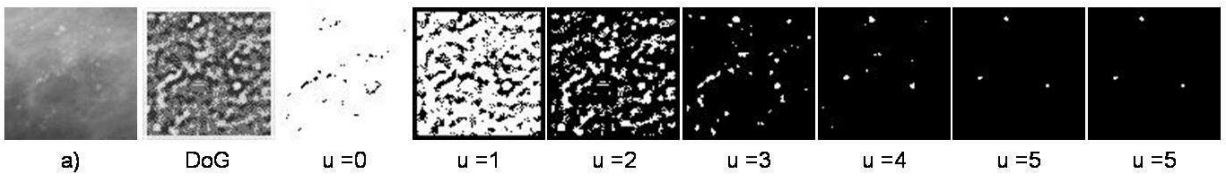


Figura 8: Resultados obtenidos para diferentes niveles del umbral. DoG(5x5) - DoG(7x7)

3.3. Etiquetado de regiones

Luego de obtener la imagen binarizada procedemos a etiquetarla, el etiquetado permite identificar aquellas regiones conectadas, se usa cuando se requiere un análisis exhaustivo de la imagen (Fontoura [9], Russ[24]). Posteriormente estas regiones se convertirán en *puntos*.

El algoritmo toma la imagen binarizada D como entrada y la recorre pixel a pixel, de izquierda a derecha y de arriba hacia abajo, en cada pixel toma una decisión sobre el color a asignarle dependiendo de una mascara de tres pixeles conexos (ver 9), buscando todos los puntos conectados en vecindad V_4. El número de colores obtenido es igual al número de regiones en la imagen.

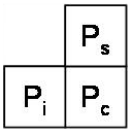


Figura 9: Mascara usada en el algoritmo de coloreado de regiones

La variable que representa el pixel actual es P_c, el pixel ubicado encima es P_s y el pixel ubicado a la izquierda es P_i, la representación asegura la exploración exhaustiva. Si P_c es un pixel del fondo (0), no se realiza ningún cálculo, si P_c es un pixel que pertenece a un objeto (1) se toma una decisión basada en los valores de P_s y P_i, de la siguiente forma:

- Siembra nuevo color. Cuando P_s y P_i corresponden al fondo de la imagen se siembra un nuevo color , esto es que se ha encontrado un nuevo objeto, el contador de colores se incrementa en 1.
- Propagación lateral del color. Cuando el pixel superior es fondo y el pixel de la izquierda es objeto, se propaga el color del objeto al pixel actual.
- Propagación vertical del color. Cuando el pixel de la izquierda es fondo y el pixel superior es objeto, se propaga el color del objeto al pixel actual.
- Cruce de regiones. Cuando el ambos pixeles superior e izquierdo son objeto, se verifica sus colores, si son iguales, se propaga alguno de ellos, si son diferentes se aplica un algoritmo para unir regiones predominando el color de menor valor relativo y asignando este nuevo color al pixel actual.

3.4. Selección de puntos por área mínima

Este procedimiento elimina aquellas regiones que caen fuera de los rangos de área mínima y máxima y luego calcula el centro de masa de las regiones seleccionadas. El procedimiento recibe como entradas una imagen que contiene regiones con pixeles conectados *color* indicando el color que le corresponde y la imagen binarizada D, se obtiene como resultado listado de los centros de masa de cada una de las regiones P y la imagen binarizada D luego de ser depurada, tiene dos argumentos el $\acute{a}rea\_m\acute{i}nima = 2$ y el $\acute{a}rea\_m\acute{a}xima = 77$. Aquellas regiones seleccionadas serán consideradas como un *puntos* (ver figura 10).

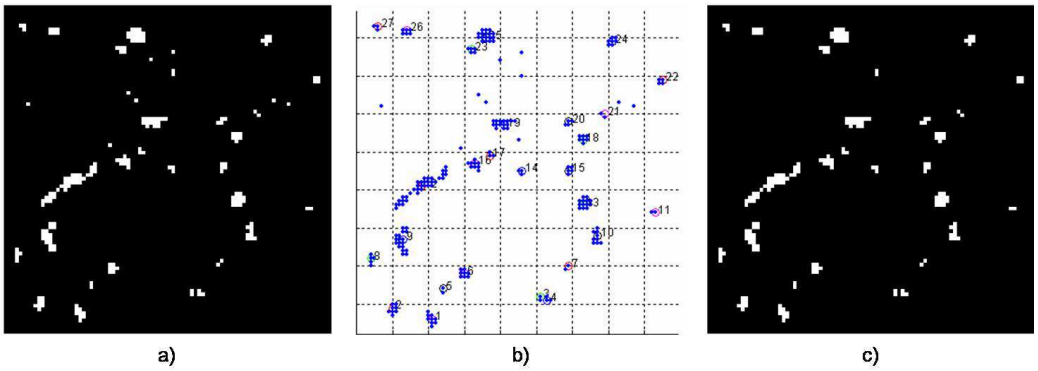


Figura 10: Etiquetado de regiones. a) Imagen Binarizada, b) Etiquetado de Regiones, c) Regiones seleccionadas como *puntos*

3.5. Segmentación

El procedimiento de segmentación extrae ventanas cuadradas de 9x9, cuyo centro corresponde al centro de masa de cada uno de los *puntos* seleccionados en la etapa anterior. Extraer ventanas conteniendo los

puntos tiene dos finalidades, primero depurar aquellas regiones en la imagen cuyos promedios de gris sean menores que cierto umbral y segundo, aplicar una binarización local que permitirá obtener finalmente las *señales* que tienen alta probabilidad de ser *microcalcificaciones*.

El procedimiento recibe como entradas la imagen pre-procesada I, la imagen binarizada D, el listado de los centros de masa de cada región P, se obtiene como resultado la lista de *puntos* tomados de la imagen pre-procesada MI, lista de *puntos* tomados de la imagen binarizada MD y tiene un argumento el tamaño del lado de la ventana *tamaño_lado*.

El algoritmo para cada *punto*, corta en cada imagen un cuadrado cuyo centro es el centroide P de lado *tamaño_lado*.

3.6. Selección de puntos por gris mínimo

Este procedimiento permite seleccionar aquellos *puntos* que cumplen el criterio de promedio de gris mínimo, en este caso se ha colocado en 50, en una escala de [0, 255].

El procedimiento recibe como entradas el listado de los centros de masa de cada *punto* P, el listado de *puntos* tomados de la imagen pre-procesada MI, el listado de *puntos* tomados de la imagen binarizada MD, se obtiene las mismas salidas y tiene un argumento el el gris mínimo.

El área considera para el cálculo del promedio en cada ventana de MI depende de su mascara asociada MD (ver figura 11)

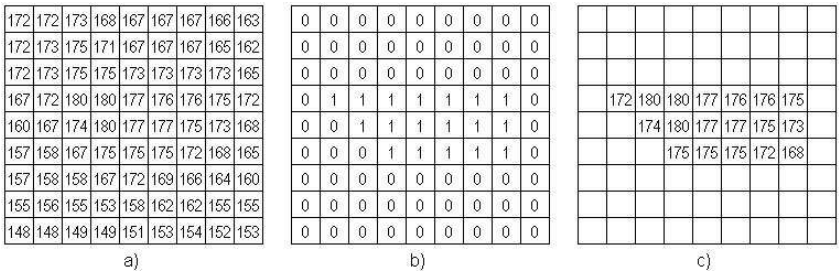


Figura 11: Calculo del gris promedio. a). niveles de gris (MI), b). mascara de la región de interés (MD), c). región de interés. Promedio total = 166, Promedio ROI = 175

El algoritmo depura aquellos puntos con promedio de gris menores que el gris mínimo, eliminándolos de las lista P, MI y MD.

3.7. Binarizacion local

El procedimiento de binarización local se aplica a cada una de las ventanas resultantes MI, con la finalidad de obtener las *señales* que luego pasarán al proceso de clasificación, bajo el supuesto de que histograma del *punto* seleccionado se puede dividir en dos regiones, el fondo y la *señal*.

El algoritmo de Otsu [22] es usado para calcular el umbral de binarización automáticamente, bajo el supuesto que el histograma es bimodal, el algoritmo consiste en buscar el mejor umbral T que separe las dos modas del histograma.

En la figura 12, se presenta dos casos en el que aplicamos el procedimiento de calculo de umbral, en el gráfico superior se presenta el histograma y la distribución de la función a maximizar *omega* (ver Otsu [22]), se traza una línea donde *omega* es máximo, se puede apreciar como que la línea trazada divide al histograma en dos campanas, correspondientes a las modas. Debajo de cada gráfica, presentamos 3 ventanas: la primera corresponde a la imagen pre-procesada, la segunda corresponde al *punto* y la tercera corresponde a la *señal* obtenida por este procedimiento.

4. EXTRACCIÓN DE CARACTERÍSTICAS

Esta etapa tiene por finalidad extraer características desde cada *señal* identificada en la etapa anterior, para luego clasificarlas como *calcificación* o como *no calcificación*. En posteriores trabajos estas características pueden ser usados para clasificar clusters de microcalcificaciones.

En total 5 características fueron extraídas: área, nivel de gris promedio, nivel promedio de gris del fondo, contraste relativo y compacidad

1. Área.
   Calcula el área que ocupa la *señal* (número de pixels).

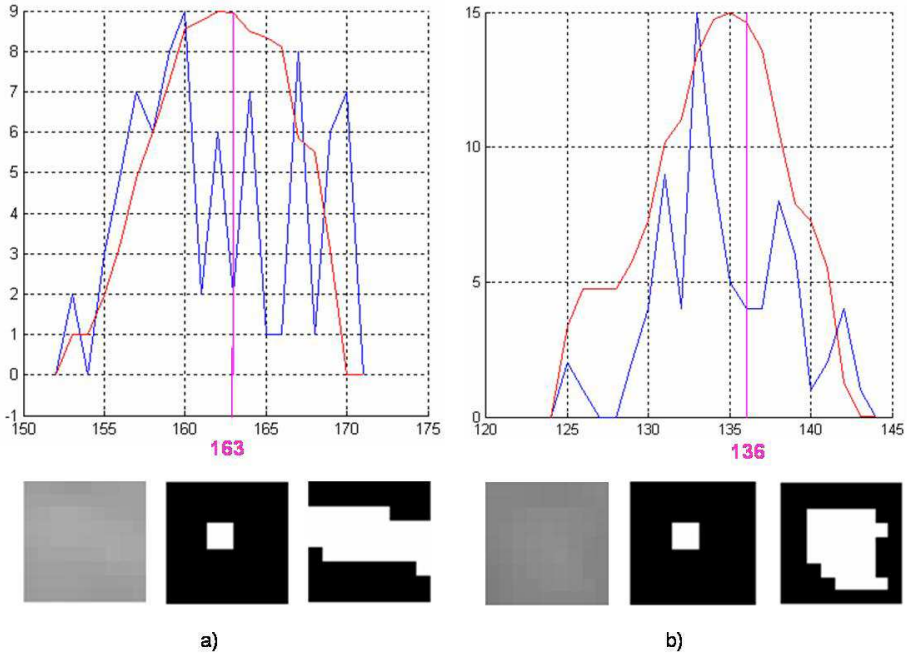


Figura 12: Ejemplos de cálculo del umbral de binarización local

$$área = \sum_{f=1}^{f=N} \sum_{c=1}^{f=N} D(f,c) \tag{2}$$

Donde $N = 9$ es el lado de la ventana y $D(f,c)$ es un pixel de la *señal* binarizada.

2. Nivel promedio de gris.
   Obtiene el promedio de los niveles de gris de la *señal*, es un buen indicador del nivel de brillo.

$$promedio\_señal = \frac{1}{área} \sum_{i=1}^{i=N} D(f,c)I(f,c) \tag{3}$$

   Donde: $D(f,c)$ es un pixel de la *señal* binarizada y $I(f,c)$ es nivel de gris del pixel.

3. Nivel promedio de gris del fondo.
   Obtiene el promedio de los niveles de gris del fondo.

$$promedio\_fondo = \frac{1}{(N^2 - área)} \sum_{i=1}^{i=N} (1 - D(f,c))I(f,c) \tag{4}$$

4. Contraste relativo.
   El contraste relativo se calcula con el promedio de gris de la calcificación y con el promedio de gris del fondo, proporciona una medida de contraste.

$$contraste\_relativo = \frac{promedio\_señal - promedio\_fondo}{promedio\_señal + promedio\_fondo} \tag{5}$$

5. Compacidad.

   La compacidad de una medida de qué tan compacta o aglutinada es la *señal*.

$$compacidad = \frac{perímetro^2}{área} \tag{6}$$

   Los objetos que con bordes irregulares y por lo tanto grandes tendrán un valor alto de compacidad.

   El menor valor que puede tomar la *compacidad* es 4π en el caso de un círculo perfecto.

5. CLASIFICACIÓN

En este trabajo usamos una red neuronal de tres capas con conexión hacia adelante usando una algoritmo de retro-propagación. La data de entrada y la data de salida fueron normalizadas antes de ser usadas en el proceso de entrenamiento. La red fue entrenada en 200 épocas.

5.1. Redes Neuronales Artificiales

Las redes neuronales artificiales han sido usadas en múltiples problema de clasificación de patrones, donde se requiere aprender de la experiencia, de generalizar casos o de abstraer características esenciales a partir de información irrelevante. La red está conformada por muchos elementos computacionales (nodos) no lineales que operan en paralelo. Los nodos están conectados en capas mediante pesos que son adaptados en el proceso de entrenamiento. El algoritmo de retro-propagación es usado para entrenar los pesos de la red en dos fases, en la primera un patrón de entrenamiento es presentado propagándose a través de la red hasta la salida, donde se calcula el error (salida deseada Vs. salida obtenida); en la segunda fase estos errores se trasmiten hacia atrás, hacia los nodos de la capa de entrada, recibiendo cada nodo un porcentaje del error. Basado en este error es que se ajustan los pesos de los nodos.

5.2. Arquitectura de la Red

Una red neuronal de retro-propagación de tres capas (ver figura 13), es usada en este trabajo, la red neuronal tiene tantas entradas como características tiene cada *señal*, la capa intermedia tiene 4 neuronas, la capa de salida tiene una neurona que indica si la *señal* corresponde o no a una *calcificación*. La función de transferencia usada en todos los nodos es la *tangente hiperbólica sigmoidal* y la función de medida de performance es el *error cuadrático medio*.

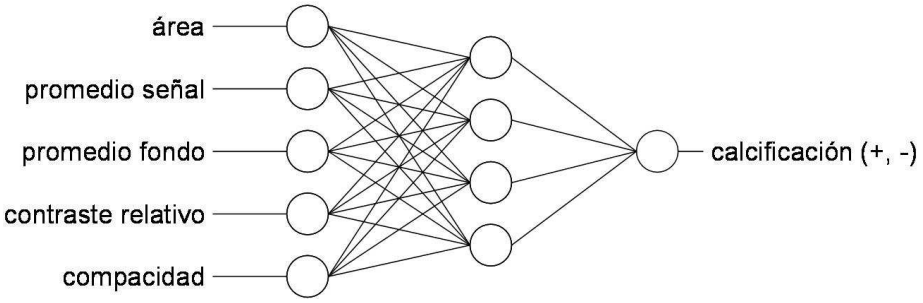


Figura 13: Arquitectura de la red neuronal

5.3. Normalización de la data

Dado que los datos utilizados como entrada a la red neuronal pueden cubrir diferentes rango de valores, sus valores fueron normalizados en el rango $[-1,\ 1]$ antes de entrar al proceso de entrenamiento, para ello se usó una tabla de máximos y mínimos. Luego durante el proceso de prueba los datos de entrar también fueron normalizados usando la tabla, los datos de salida no fueron normalizados, dado que solo pueden tomar dos valores $[-1,\ +1]$.

6. RESULTADOS EXPERIMENTALES

Para estimar el indicador de efectividad del procedimiento propuesto, usaremos el estimador por validación cruzada de K conjuntos (Efron [11], Stone [26]), usado frecuentemente para probar esquemas de clasificación.

Se usaron 50 imágenes de 322 imágenes de la base de datos del MIAS, 25 de ellas con calcificaciones y 25 de ellas sin calcificaciones.

El diseño consiste en armar 10 experimentos, primero las 50 imágenes son divididas aleatoriamente en 10 conjuntos disjuntos conformados por 5 imágenes cada uno, luego cada experimento se construye tomando el primer conjunto como prueba y los 9 siguientes como entrenamiento, para el segundo experimento se toma el segundo conjunto como prueba y los restantes 9 conjuntos como entrenamiento, así hasta armar los 10 experimentos.

El detalle de los resultados se muestran en la tabla 1. Ellos corresponden a los mejores resultados obtenidos en promedio luego de sucesivas pruebas con el clasificador. El promedio de eficiencia es del 62.45 %.

Cuadro 1: Resultados Obtenidos

Grupo	Porcentaje de éxito
1	50.00
2	59.38
3	56.25
4	56.25
5	75.76
6	50.00
7	71.88
8	53.13
9	56.25
10	42.42
Promedio	62.45
STD	7.99

7. CONCLUSIONES Y RECOMENDACIONES

Los resultados experimentales obtenidos hasta el momento son del 62.45 % de eficacia, estos resultados se pueden mejorar en la medida que los parámetros usados en el sistema se optimicen.

El número de casos de entrenamiento es relativamente poco para obtener resultados determinantes, este problema se encuentra en proceso de solución. Se trata de ampliar el numero de muestras considerando las señal como muestras y no las imágenes como muestras, lo cual permitiría desde una imagen extraer un número grande de muestras.

En este momento estamos probando otros clasificadores para intentar mejorar la eficacia del sistema, tales como árboles de decisión y reglas de asociación.

Reconocimientos

El autor agradece al Dr. Hugo Terashima y al MSc. Rolando Hernández Centro de Sistemas Inteligentes, Instituto Tecnológico de Monterrey ITESM, por sus aportes y sugerencias en el desarrollo de este proyecto.

Referencias

[1] R. Brancaccio R. Campanini N. Lanconelli A. Bazzani, D. Bollini and D. Romani. System for automatic detection of clustered microcalcifications in digital mammograms. *International Journal of Modern Physics C*, 11(5):901–912, 2000.

[2] I. Anttinen, M. Pamilo, M. Soiva, and M. Roiha. Double reading of mammography screening films: one radiologist or two? *Clin Radiol*, (48):414–421, 1993.

[3] L. Bocchi, G. Coppini, J.Ñori, and G. Valli. Detection of single and clustered microcalcifications in mammograms using fractals models and neural networks. *Med Eng Phys*, 26(4):303–12, 2004.

[4] R. Campanini, A. Bazzani, A. Bevilacqua, D. Bollini, D.N. Dongiovanni, E. Iampieri, N.Lanconelli, A. Riccardi, M. Roffilli, and R. Tazzoli. A novel approach to mass detection in digital mammography based on support vector machines (svm). In *Proc. of IWDM2002*, pages 399–401, Bremen, Germany, June 22-25, 2002, 2002.

[5] C.A. Castro, T.S. Pérez, J.M. González Barcena, and J. R. Santiago. Interferón y cáncer de mama avanzado. *Revista Cubana de Oncología*, 15(2):89–94, 1999.

[6] H. D. Cheng, Y.M. Lui, and R. I. Freimanis. A novel approach to microcalcification detection using fuzzy logic technique. *IEEE Transactions on Medical Imaging*, 17(3):442–450, 1998.

[7] S. Ciatto, MR. Del Turco, G. Risso, S. Catarzi, R. Bonardi, V. Viterbo, P. Gnutti, B. Guglielmoni, L. Pinelli, A. Pandiscia, F.Ñavarra, A. Lauria, R. Palmiero, and PL. Indovina PL. Comparison of standard reading and computer aided detection (cad) on a national proficiency test of screening mammography. *European Journal of Radiology*, 45(2):135–8, 2003.

[8] F. Calero Cuerda. Factores de riesgo en cancer de mama. *Progresos de Obstetricia y Ginecología*, 42(90):9065–9088, 1999.

[9] L. da Fontoura and R. Marcondes. *Shape Analysis and Classification. Theory an Practice*. CRC Press, 2001.

[10] S. De Paredes E. *Radiographic breast anatomy: Radiologic signs of breast cancer, Syllabus: a categorical course in physics - Technical aspects of breast imaging*. M Yaffe ed., Oak Brook, IL, RSNA Publications, 1993.

[11] B. Efron and R. Tibshirani. An introduction to the bootstrap. *Journal of the National Cancer Institute*, 94(18):1373–80, 2002.

[12] R. Egan. *Breast Imaging: Diagnosis and Morphology of Breast Diseases*. Philadelphia. W.B. Saunders Company, 1988.

[13] I. El-Naqa, Y. Yang, M.Ñ. Wernick, N. P. Galatsanos, and R. M. Nishikawa. A support vector machine approach for detection of microcalcifications. *IEEE Transactions on Medical Imaging*, 21(12):1552–1563, 2002.

[14] J.G. Elmore, D. L. Miglioretti, L. M. Reisch, M. B. Barton, W. Kreuter, C. L. Christiansen, and S. W. Fletcher. Screening mammograms by community radiologists:variability in false-positive rates. *Journal of the National Cancer Institute*, 94(18):1373–80, 2002.

[15] Marie A. Ganott, MD, Kathleen M. Harris, MD, FACR, Herta M. Klaman, RT(R)(N)(M), RDMS, RDCS, and Terri L. Keeling. Analysis of false-negative cancer cases identified with a mammography audit. *The Breast Journal*, 5(3):166, 1999.

[16] J.M. Alonso Gordo. Cancer de mama. manejo desde atención primaria. *SEMERGEN*, 26:491–501, 2000.

[17] S. Hayken. *Neural networks: A comprehensive foundation*. Nueva York: Macmillan, 1994.

[18] Norhayati Ibrahim, Hiroshi Fujita, Takeshi Hara, and Tokiko Endo. Automated detection of clustered microcalcifications on mammograms: Cad system application to mias database. *Physics in Medicine and Biology*, 42(12):2577–2589, 1997.

[19] N. Karssemeijer and G. Brake. Detection od stellate distortions in mammograms. *IEEE Transactions on Medical Imaging*, 15(5):611, 1996.

[20] MIAS. http://www.wiau.mam.ac.uk/services/mias/miasweb.html. 2002.

[21] AI. Mushlin, RW. Kouides, and DE. Shapiro. Estimating the accuracy of screening mammography: a meta-analysis. *Am J Prev Med*, 14(2):143–53, 1998.

[22] N. Otsu. A thresholding selection method from gray-level histogram. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[23] R. M. Rangayyan, N. M. El-Faramawy, J.E. Leo Desautels, and O. A. Alim. Measures of acurate and shape for classification of breast tumors. *IEEE Transactions on Medical Imaging*, 16(6):799–810, 1997.

[24] J. C. Russ. *The Image Processing Handbook Third Edition*. CRC Press, IEEE Press, 1999.

[25] Y. Shen and M. Zelen. Screening sensitivity and sojourn time from breast cancer early detection clinical trials. mammograms and physical examinations. *Journal of Clinical Oncology*, 19(15):3490–9, 2001.

[26] M. Stone. Cross-validation choice and assessment of statistical predictions. 1993.

[27] EL. Thurfjell, KA. Lernevall, and AAS. Taube. Benefit of independent double reading in a population-based mammography screening program. *Radiology*, (191):241–244, 1994.

[28] UCHC. radiology.uchc.edu. 2004.

[29] Songyang Yu and Ling Guan. A cad system for the automatic detection of clustered microcalcifications in digitized mammograms films. *IEEE Transactions on Medical Imaging*, 19(2):115–126, 2000.

Omicron ACO

Osvaldo Gómez
Universidad Nacional de Asunción
Centro Nacional de Computación
Asunción, Paraguay
ogomez@cnc.una.py

and

Benjamín Barán
Universidad Nacional de Asunción
Centro Nacional de Computación
Asunción, Paraguay
bbaran@cnc.una.py

Abstract

Ant Colony Optimization (ACO) is a metaheuristic inspired by the foraging behavior of ant colonies that has been successful in the resolution of hard combinatorial optimization problems like the TSP. This paper proposes the Omicron ACO (OA), a novel population-based ACO alternative designed as an analytical tool. To experimentally prove OA advantages, this work compares the behavior between the OA and the $\mathcal{MMAS}$ as a function of time in two well-known TSP problems. A simple study of the behavior of the OA as a function of its parameters proves its robustness.

Keywords: Artificial Intelligence, Ant Colony Optimization, Omicron ACO, $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System.

1 Introduction

Ant Colony Optimization (ACO) is a metaheuristic proposed by Dorigo et al. that has been inspired by the foraging behavior of ant colonies [3]. In the last few years ACO has empirically shown its effectiveness in the resolution of several different NP-hard combinatorial optimization problems; however, still little theory is available to explain the reasons underlying ACO's success. Birattari et al. developed a formal framework of ant programming with the goal of gaining deeper understanding on ACO [1], while Meuleau and Dorigo studied the relationship between ACO and Stochastic Gradient Descent [10]. Gutjahr presented a convergence proof for a particular ACO algorithm called Graph-based Ant System (GBAS) that has an unknown empirical performance [7]. He proved that the GBAS converges, with a probability that could be made arbitrarily close to 1, to the optimal solution of a given problem instance. Later, Gutjahr demonstrated for a time-dependent modification of the GBAS that its current solutions converge to an optimal solution with probability exactly equal to 1 [8]. Stützle and Dorigo presented a short convergence proof for a class of ACO algorithms called $\text{ACO}_{gb,\tau_{min}}$ [11], where *gb* indicates that the global best pheromone update is used, while τ_{min} indicates that a lower limit on the range of the feasible pheromone trail is forced. They proved that the probability of finding the optimal solution could be made arbitrarily close to 1 if the algorithm is run for a sufficiently large number of iterations.

In search of new ACO analytical tools, a simple algorithm preserving certain characteristics of ACO was developed. This is how the Omicron ACO (OA) was conceived and its name comes from the main parameter used, which is *Omicron* (*O*). OA was motivated by the ideas behind ACO, i.e. the search for nearby good solutions. OA was first designed with theoretical motivations to study convergence properties [4], but proved to be very useful also in practical applications. Therefore, this paper compares OA with respect to one of the best-known ACO algorithms, the $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System ($\mathcal{MMAS}$) proposed by Stützle and Hoos [12]. OA is a more straightforward utilization of the successful reasons of ACO. As a consequence, OA outperforms $\mathcal{MMAS}$ in the preliminary experimental results shown in this work. Besides, its simplicity and decreased sensibility to input parameters make it easy to configure.

This paper is organized as follows. In Section 2 the test problem and the definitions are presented. The standard ACO approach, the ideas that motivated OA and its pseudocode are given in Section 3. In Section 4, a performance comparison between $\mathcal{MMAS}$ and OA, and a simple study of the OA as a function of its parameters are made. Experimental results are explained in Section 5. Finally, the conclusions and future work are presented in Section 6 .

2 Test Problem

In this paper the symmetric Traveling Salesman Problem (TSP) is used as a test problem for comparing the analyzed algorithms. The TSP is a hard combinatorial optimization problem, easy to understand, which has been considerably studied by the scientific community. Researchers have applied ACO successfully to this problem [3, 12]. To make performance comparisons, standard TSP instances extracted from TSPLIB[1] library have been used in this work. The TSP can be represented by a complete graph $G = (N, A)$ with N being the set of nodes, also called cities, and A being the set of arcs fully connecting the nodes. Each arc (i, j) is assigned a value $d(i, j)$ which represents the distance between cities i and j. The TSP is the problem of finding the shortest closed tour visiting each of the $n = |N|$ nodes of G exactly once. For symmetric TSPs, the distances between the cities are independent of the direction of traversing the arcs, that is $d(i, j) = d(j, i)$ for every pair of nodes. Suppose that r_x and r_y are TSP tours or solutions over the same set of n cities. For this work, $l(r_x)$ denotes the length of tour r_x. The distance between r_x and r_y is defined as n minus the number of edges contained in both r_x and r_y.

3 Ant Colony Optimization

Ant Colony Optimization (ACO) is a metaheuristic inspired by the behavior of ant colonies [3]. In the last few years, ACO has received increased attention by the scientific community as can be seen by the growing number of publications and the different fields of application [12]. Even though there exist several ACO variants, what can be considered a standard approach is next presented [5].

3.1 Standard Approach

ACO uses a pheromone matrix $\tau = \{\tau_{ij}\}$ for the construction of potential good solutions. The initial values of τ are set $\tau_{ij} = \tau_{init}\ \forall(i, j)$, where $\tau_{init} > 0$. It also takes advantage of heuristic information using

[1] Accessible at http://www.iwr.uni-heidelberg.de/iwr/comopt/software/TSPLIB95/

$\eta_{ij} = 1/d(i,j)$. Parameters α and β define the relative influence between the heuristic information and the pheromone levels. While visiting city i, $\mathcal{N}_i$ represents the set of cities not yet visited. The probability of choosing a city j at city i is defined as

$$\mathcal{P}_{ij} = \begin{cases} \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{\forall g \in \mathcal{N}_i} \tau_{ig}^{\alpha} \cdot \eta_{ig}^{\beta}} & \text{if } j \in \mathcal{N}_i \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

At every generation of the algorithm, each ant of a colony constructs a complete tour using (1), starting at a randomly chosen city. Pheromone evaporation is applied for all (i,j) according to $\tau_{ij} = (1-\rho) \cdot \tau_{ij}$, where parameter $\rho \in (0,1]$ determines the evaporation rate. Considering an elitist strategy, the best solution found so far r_{best} updates τ according to $\tau_{ij} = \tau_{ij} + \Delta\tau$, where $\Delta\tau = 1/l(r_{best})$ if $(i,j) \in r_{best}$ and $\Delta\tau = 0$ if $(i,j) \notin r_{best}$. For one of the best performing ACO algorithms, the $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System ($\mathcal{MMAS}$) [12], minimum and maximum values are imposed to τ (τ_{min} and τ_{max}).

3.2 Omicron ACO

OA was initially inspired by $\mathcal{MMAS}$, an ACO currently considered among the best performing ACOs for the TSP [12]. It is based on the hypothesis that it is convenient to search for nearby good solutions [2, 12].

The main difference between $\mathcal{MMAS}$ and OA is the way the algorithms update the pheromone matrix. In OA, a constant pheromone matrix τ^0 with $\tau_{ij}^0 = 1$, $\forall i,j$ is defined. OA maintains a population $P = \{P_x\}$ of m individuals or solutions, the best unique ones found so far. The best individual of P at any moment is called P^*, while the worst individual P_{worst}.

In OA the first population is chosen using τ^0. At every iteration a new individual P_{new} is generated, replacing $P_{worst} \in P$ if P_{new} is better than P_{worst} and different from any other $P_x \in P$. After K iterations, τ is recalculated. First, $\tau = \tau^0$; then, O/m is added to each element τ_{ij} for each time an arc (i,j) appears in any of the m individuals present in P. The above process is repeated every K iterations until an end condition is reached (see pseudocode for details). Note that $1 \leq \tau_{ij} \leq (1+O)$, where $\tau_{ij} = 1$ if arc (i,j) is not present in any P_x, while $\tau_{ij} = (1+O)$ if arc (i,j) is in every $P_x \in P$.

Similar population-based ACO algorithms (P-ACO) [5, 6] were designed by Guntsch and Middendorf for dynamic combinatorial optimization problems. The main difference between the OA and the *Quality* Strategy of P-ACO is that OA does not allow identical individuals in its population. Also, OA updates τ every K iterations, while P-ACO updates τ every iteration.

Next, the pseudocode of the OA considering a TSP with n cities is presented.

Pseudocode of the main *Omicron ACO*

```
   Input parameters: n, matrix D = {d_ij}, O, K, m, α, β
   Output parameter: P  (m best found solutions)

τ = Initialize the pheromone matrix ()
P = Initialize the population (τ)
REPEAT UNTIL end condition
   REPEAT K TIMES
      P_new = Construct a solution (τ)
      IF (length(P_new) < length(P[0])) and (P_new ≠ all elements of P)
         P = Update population (P_new , P)
   τ = Update pheromone matrix (P)
```

Pseudocode of the function *Initialize the pheromone matrix ()*

```
REPEAT for every arc (i,j)
      τ[i,j] = 1
```

Pseudocode of the function *Initialize the population (τ)*

```
x = 0
WHILE x < m
   P_new = Construct a solution (τ)
   IF  P_new ≠ all chosen elements of P
      P[x] = P_new
      x = x + 1
P = Sort P from worst to best considering the tour length   /*  P_worst = P[0]  */
```

Pseudocode of the function *Construct a solution (τ)*

```
P_new[0] = Select a city randomly
x = 1
WHILE x < n
   P_new[x] =  Select a city randomly considering equation (1)
   x = x + 1
```

Pseudocode of the function *Update population (P_{new} , P)*

```
P[0] = P_new
P = Sort P efficiently from worst to best considering the tour length
      /*  P_worst = P[0]  */
```

Pseudocode of the function *Update pheromone matrix (P)*

```
τ = Initialize the pheromone matrix ()
x = 0
WHILE x < m
   REPEAT for every arc (i,j) of P[x]
      τ[i,j] = τ[i,j] + O/m
   x = x + 1
```

4 Experimental Results

For the following experimental results, a 2 GHz computer with 256 MB of RAM was used. The programming language chosen was C and the operating system was Linux. First, a comparative study between $\mathcal{MMAS}$ and OA is presented. Then, the behavior of OA as a function of its parameters is studied.

4.1 Comparative Study Between OA and $\mathcal{MMAS}$

As a performance reference, the parameters for the $\mathcal{MMAS}$ algorithm were extracted from [12], where $\alpha = 1$ and $\beta = 2$ were always used and the same problems were solved. To make a fair comparison, the same parameters $\alpha = 1$ and $\beta = 2$ were also used for the OA. The rest of the parameters were found empirically, searching for a balanced behavior between the speed of convergence and the quality of the final solution, choosing $O = 600$, $m = 25$ and $K = 1,000$. Because no attempt was done to optimize OA parameters or to make them time-dependent, $\mathcal{MMAS}$ with pheromone trail smooth was not chosen for comparison.

Empirical observations were done as a function of time given that the concept of iteration or generation is not the same for both algorithms. In general, OA produces better solutions than $\mathcal{MMAS}$ from the very beginning and converges to a slightly better result. As an example, for the problem eil51 with 51 cities studied in [12], the behavior of the best solution found for each algorithm was studied. The mean of the evolution of both algorithms in 25 runs can be seen in Figure 1. In Figure 2 the ranges are modified to show the clear advantage of OA at the convergence stage.

The OA results are promising considering two reasons. First, the minimum tuning work made in the algorithm parameters and second, the fact that partial results show an increased robustness, since using exactly the same parameters similar results are observed in the 100 cities problem kroA100.

In Figure 3 we observe the mean in 25 runs of the evolution of the best length of both algorithms for a larger problem, the well known kroA100 [12]. In Figure 4 the ranges are modified, as in Figure 2, to show the advantage of OA over $\mathcal{MMAS}$, considering convergence. Once more, considering mean behavior in 25 runs, OA outperforms $\mathcal{MMAS}$.

Note that best results were obtained and bigger instances were solved using $\mathcal{MMAS}$ with local search [12]. In these preliminary tests, only little instances of the TSP were solved (because no local search was implemented) to make a comparison between both algorithms without any interference.

4.2 Simple Study of the Behavior of OA as a Function of its Parameters

To verify the robustness of OA, its behavior has been observed as a function of its parameters. In Figure 5 (a) the evolution of the mean of the best individual's length is observed; 25 runs were made using $m = 25$, $K = 1,000$ and $O = 300$, $O = 600$ and $O = 1,200$. In Figure 5 (b), the 25 runs were made using $O = 600$, $K = 1,000$ and $m = 13$, $m = 25$ and $m = 50$, while in Figure 5 (c) using $O = 600$, $m = 25$ and $K = 500$, $K = 1,000$ and $K = 2,000$.

Clearly, a significant variation of the parameters used in Section 4.1 did not alter considerably the behavior of the algorithm. Using a greater value of O, the behavior is almost identical, while a smaller value shows a

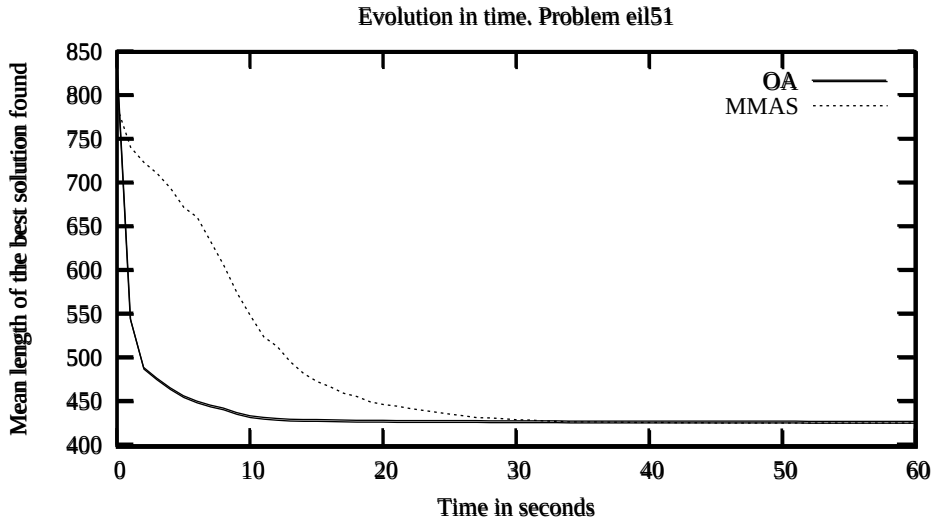


Figure 1: Comparison between the mean behavior of the proposed OA and the $\mathcal{MMAS}$

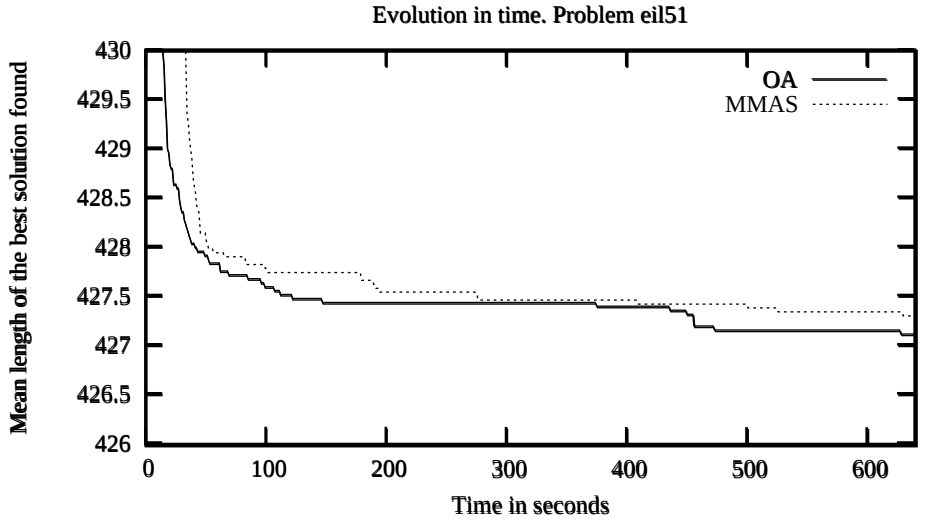


Figure 2: Comparison between the mean behavior of the proposed OA and the $\mathcal{MMAS}$ using different ranges, allowing a detailed observation of the convergence characteristics

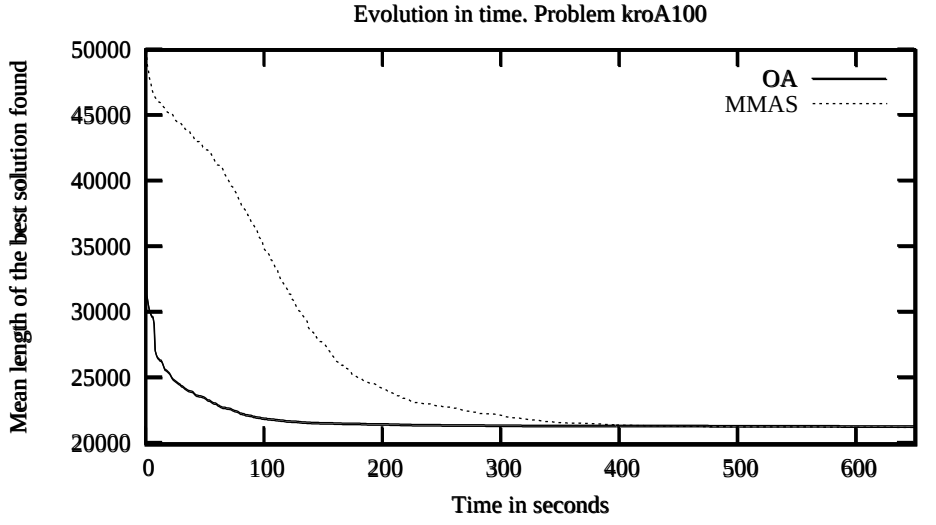


Figure 3: Comparison between the mean behavior of the proposed OA and the $\mathcal{MMAS}$

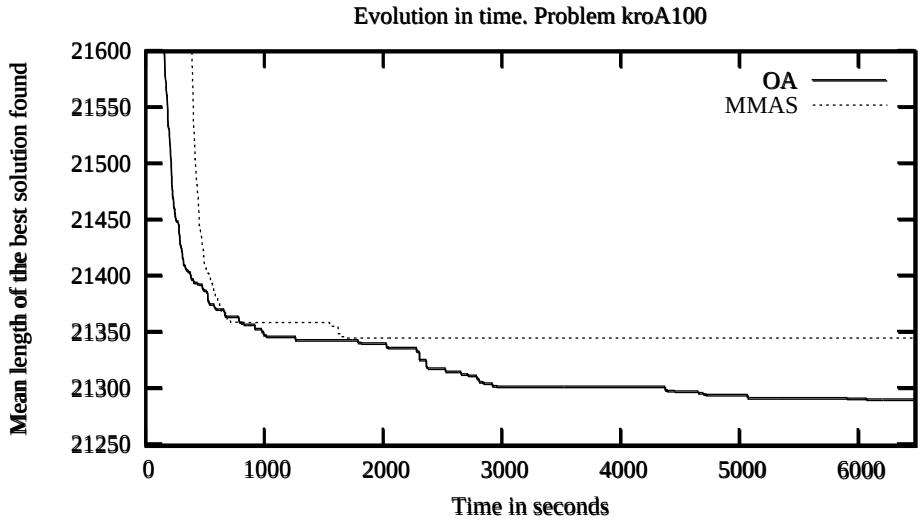


Figure 4: Comparison between the mean behavior of the proposed OA and the $\mathcal{MMAS}$ using different ranges, allowing a detailed observation of the convergence characteristics

slight fall in its performance. This is due to the diminished strength of the search for nearby good solutions. Using a larger m, the search was slower at the beginning but resulted in a slightly better solution at the end. A smaller m did the opposite. This can be seen as if the increase of the number of individuals delays the search, but ensures a good final search zone. By increasing the parameter K, a slower initial progress was observed, while decreasing it did the opposite. This can be understood because the more frequent update of the pheromone matrix allows to search for nearby better solutions in advance.

5 Explaining Experimental Results

To explain the reasons why OA outperforms $\mathcal{MMAS}$, one of the best-known ACO algorithms [12], it is useful to remember two main reasons why ACO is a good algorithm for a TSP with globally convex structure [2, 4, 9].

1. Given a population of good enough solutions, better solutions may be found with larger probability closer (considering the distance concept explained in Section 2) to good solutions than to bad ones. All ACO algorithms, including OA and $\mathcal{MMAS}$ are based on this known property.

2. Because the existence of local optimal solutions, it is better to search within a region defined by a whole population of good solutions than only close to the best-known one. Even though all ACO algorithms use this property in one way or another, OA is the one that best exploits this property in an explicit way, looking for good solutions mainly in the subspace spanned by the best m known solutions, without concentrating its search mainly around the best current solution which usually is only a local optimum.

Therefore, the main reason OA outperforms $\mathcal{MMAS}$ is its ability to search explicitly in a whole subspace Ω spanned by m good solutions, instead of mainly increasing the amount of pheromone of only one good solution. At the beginning, this property makes OA converge faster to a good region, given that it uses more information of each cycle (up to m pheromone updates, instead of just one). At the end, OA searches for the best solution close to a whole subspace Ω, without giving more importance to any of the m individuals of population P. On the contrary, other ACOs as $\mathcal{MMAS}$ look for new solutions mainly near the best-known solution, which usually is only a local optimum.

Finally, it may be noticed that OA completely forgets old solutions that are not members of the population, while $\mathcal{MMAS}$ evaporates the pheromone very slowly at the arcs of these old bad solutions, slowing down its convergence.

6 Conclusions and Future Work

This paper presents Omicron ACO (OA), a new algorithm inspired by one of the best ACO algorithms, the $\mathcal{MMAS}$ [12]. OA uses one of the principles of the $\mathcal{MMAS}$ success more directly, the search for nearby good solutions in problems of combinatorial optimization like the TSP with globally convex structure of its

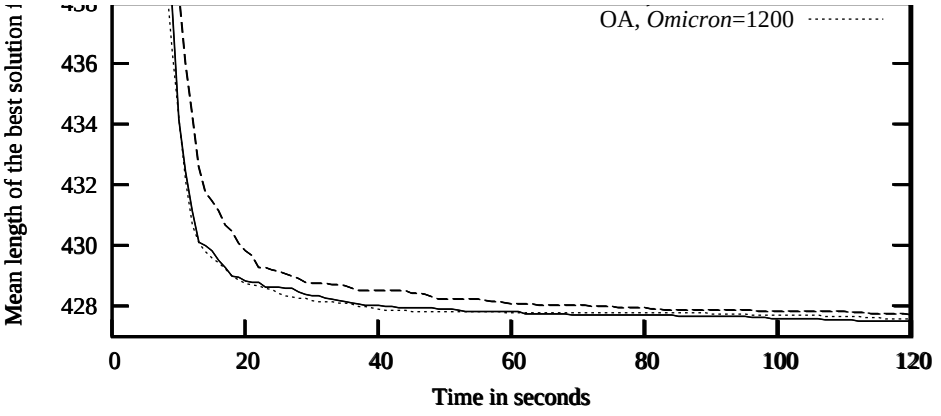


FIGURE (b). Evolution in time for different *m*

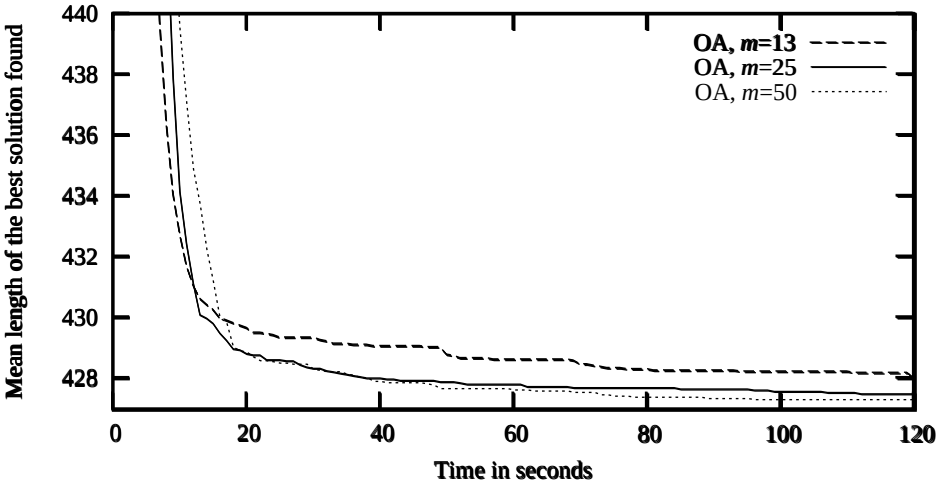


FIGURE (c). Evolution in time for different *K*

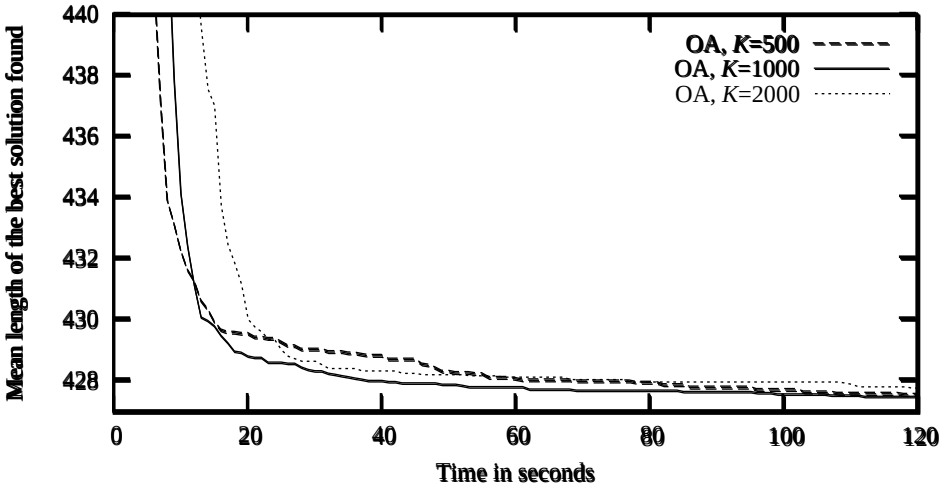


Figure 5: Comparison among the mean behavior of the proposed OA for the problem eil51, for different values of the parameters O, m and K

search space [2]. This new OA algorithm outperforms $\mathcal{MMAS}$ for these preliminary tests and it is also more robust with relation to its initial parameters; therefore, it is easier to configure.

Finally, its conceptual simplicity and the fact that it uses the same foundations as ACO allow a deeper study of the reasons of ACO's success.

After these encouraging preliminary results, the authors are working on the comparison of OA for TSPLIB problems to other ACO algorithms as P-ACO and $\mathcal{MMAS}$ with pheromone trail smooth and local search. Future work may concentrate on a deeper theoretical study of OA and its application to other combinatorial optimization problems in comparison with other world-class metaheuristics.

References

[1] M. Birattari, G. Di Caro, and M. Dorigo. For a Formal Foundation of the Ant Programming Approach to Combinatorial Optimization. Part 1: The problem, the representation, and the general solution strategy. Technical Report TR-H-301, ATR-Human Information Processing Labs, Kyoto, Japan, 2000.

[2] Kenneth D. Boese. Cost Versus Distance in the Traveling Salesman Problem. Technical Report 950018, University of California, Los Angeles, Computer Science Department, May 19, 1995.

[3] Marco Dorigo and Gianni Di Caro. The Ant Colony Optimization Meta-Heuristic. In David Corne, Marco Dorigo, and Fred Glover, editors, *New Ideas in Optimization*, pages 11–32. McGraw-Hill, London, 1999.

[4] Osvaldo Gómez and Benjamín Barán. Reasons of ACO's Success in TSP. In M. Dorigo, M. Birattari, C. Blum, L. M. Gambardella, F. Mondada, and T. Stützle, editors, *Proceedings of ANTS 2004 - Fourth International Workshop on Ant Colony Optimization and Swarm Intelligence*, volume 3172 of *LNCS*, Brussels, September 2004. Springer-Verlag.

[5] Michael Guntsch and Martin Middendorf. A Population Based Approach for ACO. In Stefano Cagnoni, Jens Gottlieb, Emma Hart, Martin Middendorf, and Günther Raidl, editors, *Applications of Evolutionary Computing, Proceedings of EvoWorkshops2002: EvoCOP, EvoIASP, EvoSTim*, volume 2279, pages 71–80, Kinsale, Ireland, 3-4 2002. Springer-Verlag.

[6] Michael Guntsch and Martin Middendorf. Applying Population Based ACO to Dynamic Optimization Problems. In *Ant Algorithms, Proceedings of Third International Workshop ANTS 2002*, volume 2463 of *LNCS*, pages 111–122, 2002.

[7] Walter J. Gutjahr. A graph-based Ant System and its convergence. *Future Generation Computer Systems*, 16(8):873–888, June 2000.

[8] Walter J. Gutjahr. ACO Algorithms with Guaranteed Convergence to the Optimal Solution. *Information Processing Letters*, 82(3):145–153, May 2002.

[9] T. C. Hu, Victor Klee, and David Larman. Optimization of globally convex functions. *SIAM Journal on Control and Optimization*, 27(5):1026–1047, September 1989.

[10] Nicolas Meuleau and Marco Dorigo. Ant colony optimization and stochastic gradient descent. *Artificial Life*, 8(2):103–121, 2002.

[11] T. Stützle and M. Dorigo. A Short Convergence Proof for a Class of Ant Colony Optimization Algorithms. *IEEE Trans. on Evol. Comput.*, 6:358–365, August 2002.

[12] Thomas Stützle and Holger H. Hoos. $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System. *Future Generation Computer Systems*, 16(8):889–914, June 2000.

Exploração de *Design Rationale* de Artefatos de Software na *Web* - Um Mecanismo de Busca em Documentos XML

Lisandra C. Fumagalli
lisandra@icmc.usp.br

Renata P. M. Fortes
renata@icmc.usp.br

Universidade de São Paulo – Departamento de Computação e Estatística
São Carlos - São Paulo - Brasil CEP 13560-970

Abstract

Design Rationale (DR) consist of a set of information related to the development and decision process of a project. In software projects, acquiring and making such information available are important practices for the improvement of the development. Consequently the product must be produced with higher quality. With the documentation about the artefacts produced during the software development is possible to create links with their corresponding Design Rationale. Thus, an XML document may be considered an appropriate mechanism for the documentation activity. However, the effective use of the information contained in this XML document is only possible if these information are retrieved and explored according to the developers interests, making their search and discovery easier. The DocRationale tool was developed to allow the storage and recovery of information related to software projects and their respective DR. However, as only the simple navigation was foreseen initially, searching for DR information is very onerous. In this article we present a mechanism for DR searching as a way assist the exploration of these information.
Keywords: Design Rationale, Software Documentation, XML Document, Searching Mechanism.

Resumo

As razões de projeto (*Design Rationale* - DR) consistem em um conjunto de informações relacionadas ao processo de desenvolvimento e de tomada de decisão de um projeto. Em especial, nos projetos de software, adquirir e disponibilizar tais informações são práticas importantes para a melhoria das atividades de desenvolvimento e conseqüentemente da qualidade do produto desenvolvido. Por meio da atividade de documentação, os artefatos produzidos durante o projeto de software constituem a base para que ligações possam ser inseridas e expressem as relações com o Design Rationale correspondente. Assim, um documento XML se apresenta como mecanismo apropriado para essa atividade. No entanto, a utilização efetiva das informações nesse documento XML só é possível se elas forem recuperadas e exploradas de forma a atender às necessidades dos desenvolvedores, facilitando-se sua busca e descoberta. A ferramenta DocRationale foi desenvolvida para permitir o armazenamento e recuperação de informações de projeto de software, e respectivo DR. No entanto, para a exploração do DR armazenado, somente a navegação simples foi prevista inicialmente. Assim, a busca por informações de DR torna-se bastante custosa. Neste artigo é apresentado um mecanismo para busca de DR, de maneira a auxiliar a exploração dessas informações.

Palavras- chave: Design Rationale, Documentação de Software, Documento XML, Mecanismo de Busca.

1. INTRODUÇÃO

Na engenharia de software, o objetivo é produzir software com qualidade, reduzindo-se os custos e os esforços exigidos nas várias fases do desenvolvimento e na manutenção [30]. Nesse sentido, técnicas e métodos se consolidam e contribuem para o reaproveitamento das experiências e a retomada do processo de decisão que envolve o processo de desenvolvimento de software. Além disso, organizações de desenvolvimento de software têm empregado esforços para utilizar o conhecimento adquirido como importante recurso para o cumprimento das atividades de engenharia de software [5, 25, 35] e obter melhorias no processo de desenvolvimento. Essas ações contribuem para a melhoria da qualidade dos produtos de software.

A aquisição e recuperação de informações relacionadas às decisões de projeto se tornam, portanto, necessárias para que as experiências sejam aproveitadas em projetos futuros. Essas informações relacionadas às decisões de projeto são muito úteis para outros projetos, uma vez que erros podem ser evitados e alternativas anteriormente consideradas podem ser mais bem reaproveitadas [33], pois soluções discutidas e adotadas em um projeto podem ser relevantes a outros. Do mesmo modo, o conhecimento disponibilizado pode ser utilizado para evitar a repetição de falhas e contribuir para aquisição de familiaridade [1, 24], auxiliando os desenvolvedores na realização de suas atividades.

As razões de projeto (Design Rationale - DR) se aplicam nesse contexto, constituindo uma base de informações relacionadas às decisões tomadas e também às razões que propiciaram cada decisão, incluindo suas justificativas, alternativas consideradas, assim como o raciocínio empregado no desenvolvimento de um projeto [23, 15, 28, 31]. No caso dos projetos de software, a atividade de documentação é responsável pelo registro dos artefatos (ou seja, quaisquer tipos de documentos produzidos durante a produção do software) que são desenvolvidos. Assim, um documento XML se apresenta como mecanismo apropriado para apoiar essa atividade, propiciando que ligações sejam inseridas aos artefatos, para associar o DR correspondente.

A necessidade de armazenar as informações, tanto registradas nos diversos documentos, quanto as relacionadas a DR, e possibilitar sua efetiva disponibilização para o compartilhamento do conhecimento relacionado à execução das atividades de engenharia de software motivou o desenvolvimento da ferramenta DocRationale [13]. A DocRationale é uma ferramenta *web* que tem como finalidade permitir o armazenamento e recuperação de informações de projeto de software, e respectivo DR. No entanto, no estágio atual da ferramenta, a recuperação do DR armazenado é realizada somente através de navegação simples, tornando a busca por informações bastante custosa.

Neste artigo é apresentado um mecanismo para a busca do DR obtido e armazenado na ferramenta DocRationale de modo que a busca seja realizada de maneira mais objetiva e rápida, considerando o interesse dos desenvolvedores. Para isso são utilizadas tecnologias XML.

Na Seção 2 são apresentadas algumas estratégias investigadas como solução para a recuperação de informações de DR, considerando-se que estejam armazenadas em hiperdocumentos. As principais características da ferramenta DocRationale são apresentadas na Seção 3. O mecanismo para busca de DR adotado é descrito na Seção 4. Na Seção 5 são apresentados os trabalhos relacionados. Na Seção 6 são apresentados as conclusões e trabalhos futuros.

2. ESTRATÉGIAS DE RECUPERAÇÃO DE DR EM HIPERDOCUMENTOS

O esquema de representação de DR, ou seja, o modo com que as informações são organizadas para que sejam acessíveis, geralmente determina a forma de recuperação. No entanto, diferentes estratégias podem ser adotadas na recuperação, dependendo do interesse dos desenvolvedores [31]. Quando se considera a utilização de um hiperdocumento para esse esquema, entre as diferentes estratégias de recuperação se destaca, como mecanismo natural, a navegação. Assim, podem-se observar dois tipos: a navegação simples e a navegação adaptativa. Além da navegação, outras estratégias de recuperação de informações em hiperdocumento, que têm sido amplamente adotadas, são a pesquisa e abordagens híbridas.

2.1 Navegação Simples

A forma mais tradicional é navegação por *links*, através dos quais os nós de informação são explorados. Essa navegação pode ser realizada através de *links* entre dois ou mais documentos ou através de *links* entre partes de um mesmo documento. Existe ainda a navegação entre documentos precedentes e subseqüentes, navegação através de *links* armazenados em um histórico de documentos navegados, e navegação através de *bookmarks*, que é semelhante ao anterior, mas permite ao usuário selecionar os documentos a serem armazenados [29]. Na estratégia de navegação simples, também se enquadram as interações com os *browsers* que permitem, por exemplo, a navegação de determinada página por meio da especificação direta de seu endereço (URL) e a navegação utilizando-se os botões *back* e *forward.*

2.2 Navegação Adaptativa

A navegação adaptativa visa auxiliar os usuários a partir da adaptação de como apresentar os *links*. Por exemplo, apresentar os *links* em uma determinada ordem de classificação, apresentar ou deixar de apresentar certos *links*, agregar informações adicionais ao *links*, dentre outras [7]. No entanto, esta estratégia requer a adoção de sistemas de hipermídia adaptativa, pois são necessárias funcionalidades específicas para o suporte a adaptatividade. Os sistemas

de hipermídia adaptativa são especialmente úteis para disponibilizar informação seletiva e contextual a usuários com diferentes objetivos e níveis de conhecimento [8].

2.3 Pesquisa

Essencialmente, o objetivo da pesquisa é prover resultados de busca com qualidade, de maneira eficiente [6]. Um meio de organizar páginas w*eb* que possibilite acesso rápido a um assunto específico e as classificar por tópicos. Nesse sentido, a utilização de máquina de busca *(search engin*e) é apropriada, pois ela é capaz de relacionar em tópicos as informações pesquisadas pelos usuários [34]. A popularização e o sucesso das máquinas de busca estão relacionados ao desenvolvimento de novos algoritmos especialmente projetados para tornar mais rápida e precisa a recuperação de informações relevantes [19]. Uma máquina de busca gerencia de certa forma um índice na w*eb* e apresenta aos usuários as páginas relevantes relacionadas à pesquisa realizada. Para isso, é necessário algum tipo de esquema de classificação. A combinação entre pesquisa, índice e esquema de classificação, forma a estrutura básica de todas as máquinas de busca para w*eb* [34]. No entanto, determinar quais páginas são relevantes o suficiente para serem indexadas e ainda manter a precisão e a eficiência dessa tarefa é um enorme desafio [19]. Kobayashi e Takeda [19] classificam as pesquisas em:

- Simples: consistem na busca por uma só palavra ou frase.
- Customizadas: nessa categoria de pesquisa, a partir dos resultados obtidos em uma busca simples, procura-se refinar a mesma de modo que os melhores resultados sejam alvo de nova pesquisa.
- De diretórios: trata-se de buscas em diretórios/categorias. A habilidade de restringir a pesquisa dentro de categorias visa proporcionar melhor qualidade dos resultados.
- Por notícias atuais: nesse tipo de pesquisa, buscam-se conteúdos recentes e relevantes de fontes de alta qualidade. Os resultados podem ser apresentados eliminando ou agrupando os *links* similares. Tipicamente, essas pesquisas consideram a ordenação por data.
- Por conteúdo da *web*: nessa categoria de pesquisa, conteúdos adicionais como os de salas de bate-papo, quadros de avisos e guias regionais também são considerados. A customização também é aplicada.

Essas cinco categorias são geralmente avaliadas em termos da potencial qualidade dos dados e da facilidade de uso. No entanto, em ambientes *web*, a técnica de pesquisa por palavra-chave ainda predomina, pois é capaz de lidar efetiva e eficientemente com o imenso volume de informações contido na *web* [17].

2.4 Abordagens Híbridas

A combinação das estratégias apresentadas anteriormente caracteriza a abordagem híbrida. O principal objetivo é produzir melhores resultados na recuperação das informações.

Na ferramenta DocRationale, a estratégia de recuperação inicialmente adotada é a navegação simples. Outras características da ferramenta são apresentadas na próxima seção.

3. A FERRAMENTA DOCRATIONALE

A ferramenta DocRationale foi elaborada para permitir a aquisição, estruturação, armazenamento e recuperação de DR relacionado aos artefatos de software (todos os tipos de documentos de software, sejam na forma de diagramas, textos ou de outras mídias). Desenvolvida para a *web*, a ferramenta oferece *login* remoto e acesso controlado às informações armazenadas. Além disso, a DocRationale privilegia o fator de colaboração entre os membros das equipes de projeto para obtenção de DR.

Considerando promover uma busca satisfatória, foi adotada na DocRationale, uma combinação das perspectivas de comunicação e argumentação [32] para a aquisição e armazenamento de DR. Sob a perspectiva de comunicação, a DocRationale possibilita que diversas formas de comunicação digital (arquivos de áudio, vídeo, e-mail entre outros) sejam anexadas, complementando as informações sobre os artefatos de software. Quanto à perspectiva da argumentação, foram utilizadas características do modelo PHI (*Procedural Hierarchy of Issues*) [26] para estruturar as informações de maneira hierárquica. Na ferramenta, o modelo foi simplificado para utilizar apenas três tipos de nós: questões, posições (respostas) e argumentos. Assim, os DRs relativos aos artefatos são determinados por um conjunto de questões, posições e argumentos, obtidos na forma de anotações, que são consideradas um bom meio para aquisição de DR [33].

Com o propósito de apoiar o suporte a DR, houve a integração da DocRationale com CoTeia [3] e GroupNote [18]. CoTeia é uma ferramenta hipermídia colaborativa assíncrona de criação de páginas *web*. Possui diversas funcionalidades, sendo as mais interessantes para a DocRationale: (i) criação e edição de páginas *web*, que são feitas no próprio browser via formulário HTML (HyperText Markup Language); (ii) histórico, que permite o acesso ao conteúdo das últimas versões de cada hiperdocumento; e (iii) upload, que possibilita a submissão de arquivos de qualquer formato ao servidor CoTeia. GroupNote é um serviço aberto de anotações colaborativas na *web* implementado como uma API (*Application Programming Interface*). Suas características mais interessantes para DocRationale são: (a) compartilhamento de anotações por equipes de usuários; e (b) fornecimento da funcionalidade de hierarquia de nós.

Uma funcionalidade mais recente da DocRationale permite que um projeto de software seja criado na ferramenta através de um documento XML. Esse documento, cuja estrutura deve estar de acordo com um DTD (*Document*

Type Definition) previamente definido, contém dados sobre o projeto como, por exemplo, fases, atividades, artefatos e suas respectivas datas de início e de previsão de término. A DocRationale carrega esse documento XML e insere, automaticamente na base de dados da ferramenta, os dados nele contido sobre o projeto a ser criado.
No modelo desse documento XML apresentado na Figura 1, é possível saber que o projeto em questão possui as fases: **Definicao**, **Desenvolvimento** e **Finalizacao**. Além disso, também é possível notar que, por exemplo, a fase de **Desenvolvimento** é constituída das atividades de **Planejamento** e **Elaboracao.** Por sua vez, a atividade de **Planejamento** possui os artefatos: **Requisitos**, **Interface externa** e **Descricao do projeto**, e a atividade **Elaboracao** possui os artefatos **Codigo**, **Plano de Teste**, **Resultado do Teste**, **Procedimentos de Construcao**, **Relatorio de Problemas**, **Anotacoes** e **Revisao.**

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE fases SYSTEM "projeto1.dtd">
 <fases>

 <fase titulo="Definicao">
   <atividades>
    <atividade titulo="Especificacao do
                       trabalho">
     <artefatos>
      <artefato titulo="Enunciado">
       <data_inicio>03/05/2004</data_inicio>
       <data_prevista>10/05/2004</data_prevista>
      </artefato>
     </artefatos>
    </atividade>
   </atividades>
  </fase>

 <fase titulo="Desenvolvimento">
   <atividades>
    <atividade titulo="Planejamento">
     <artefatos>
      <artefato titulo="Requisitos">
       <data_inicio>03/05/2004</data_inicio>
       <data_prevista>10/05/2004</data_prevista>
      </artefato>
      <artefato titulo="Interface externa">
       <data_inicio>10/05/2004</data_inicio>
       <data_prevista>17/05/2004</data_prevista>
      </artefato>
      <artefato titulo="Descricao do projeto">
       <data_inicio>17/05/2004</data_inicio>
       <data_prevista>24/05/2004</data_prevista>
      </artefato>
     </artefatos>
    </atividade>
    <atividade titulo="Elaboracao">
     <artefatos>
      <artefato titulo="Codigo">
       <data_inicio>24/05/2004</data_inicio>
       <data_prevista>07/06/2004</data_prevista>
      </artefato>

      <artefato titulo="Plano de Teste">
       <data_inicio>07/06/2004</data_inicio>
       <data_prevista>14/06/2004</data_prevista>
      </artefato>
      <artefato titulo="Resultado do Teste">
       <data_inicio>14/05/2004</data_inicio>
       <data_prevista>18/05/2004</data_prevista>
      </artefato>
      <artefato titulo="Procedimentos de
                         Construcao">
       <data_inicio>24/05/2004</data_inicio>
       <data_prevista>07/06/2004</data_prevista>
      </artefato>
      <artefato titulo="Relatorio de Problemas">
       <data_inicio>24/05/2004</data_inicio>
       <data_prevista>18/06/2004</data_prevista>
      </artefato>
      <artefato titulo="Anotacoes">
       <data_inicio>24/05/2004</data_inicio>
       <data_prevista>18/06/2004</data_prevista>
      </artefato>
      <artefato titulo="Revisao">
       <data_inicio>18/06/2004</data_inicio>
       <data_prevista>25/06/2004</data_prevista>
      </artefato>
     </artefatos>
    </atividade>
   </atividades>
  </fase>

  <fase titulo="Finalizacao">
   <atividades>
    <atividade titulo="Verificacao">
     <artefatos>
      <artefato titulo="Avaliacao do trabalho">
       <data_inicio>25/06/2004</data_inicio>
       <data_prevista>30/06/2004</data_prevista>
      </artefato>
     </artefatos>
    </atividade>
   </atividades>

  </fase>

</fases>
```

Figura 1: Exemplo de documento XML para criação automática de projetos.

Os dados dos projetos, assim como os DRs relacionados aos artefatos de software, são armazenados em bases de dados relacionais, distribuídas entre a DocRationale, a CoTeia e a API GroupNote como apresentado na Figura 2.
Nas bases de dados da DocRationale são armazenados os dados sobre os projetos: fases, atividades e artefatos. Os arquivos anexos são armazenados na base de dados da CoTeia e a base de dados do GroupNote armazena os DRs dos artefatos.
Observa-se, porém, que a exploração das informações de DR pelos desenvolvedores, a partir das consultas e navegações simples habilitadas na ferramenta se mostra custosa e pouco intuitiva. De fato, a maneira como estão armazenadas as informações na DocRationale dificulta a recuperação das mesmas. Associar os dados sobre os projetos com os DRs dos artefatos relativos a cada projeto, e ainda com os arquivos anexos de cada artefato, não é uma tarefa trivial.
A aquisição de DR na DocRationale é feita de maneira estruturada de modo que a recuperação desses DRs, através de navegação simples por *links*, é definida basicamente por essa estrutura. Assim, a atividade de buscar as informações desejadas se torna muito onerosa para o desenvolvedor, que deve percorrer grande quantidade de *links* para tentar encontrar a informação desejada.

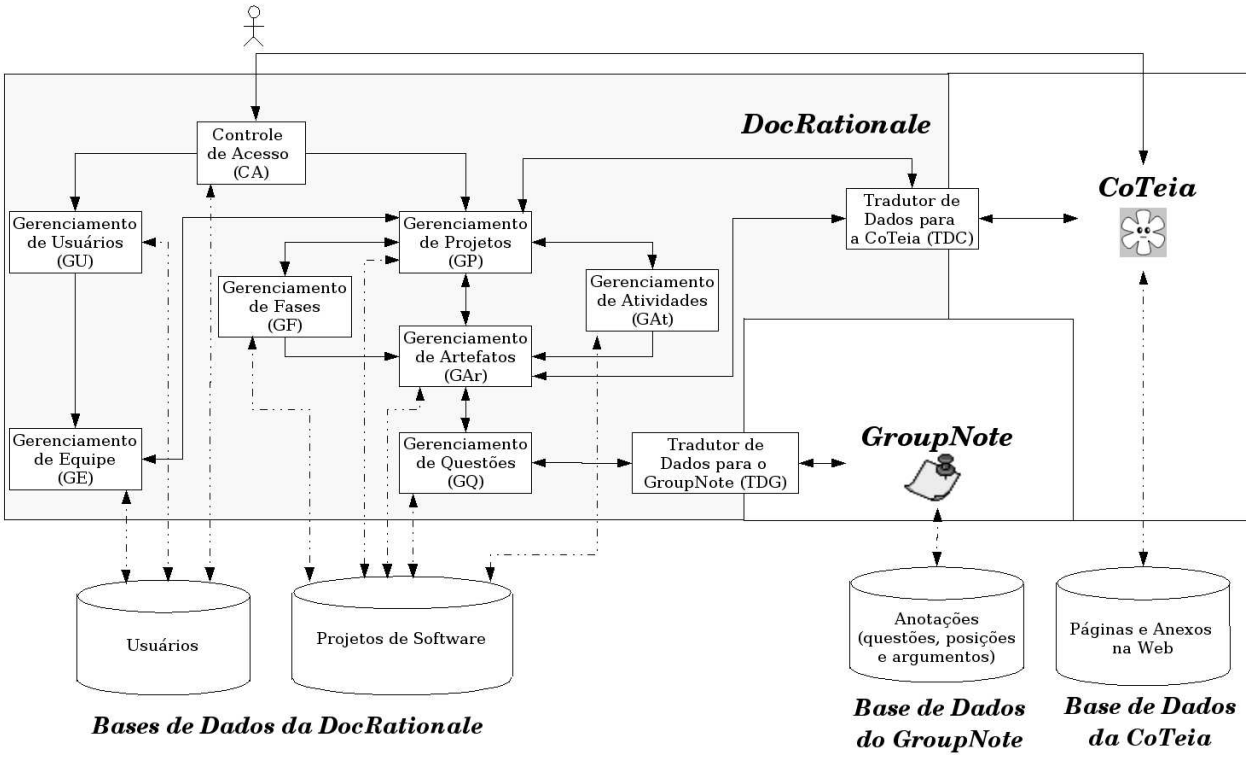


Figura 2: Arquitetura da ferramenta DocRationale.

Dessa forma, a solução adotada para resolver essa dificuldade e promover buscas mais diretas e rápidas é apresentada a seguir.

4. MECANISMO PARA A RECUPERAÇÃO DE DR

O principal objetivo é oferecer um mecanismo de busca de informações de maneira mais eficiente do que a atualmente disponível na ferramenta DocRationale, considerando os interesses dos desenvolvedores. A idéia é integrar os dados armazenados nas diferentes bases de dados de maneira a agregá-los em um único documento para cada projeto. Nesse contexto, a tecnologia *web* provê infra-estrutura apropriada, pois incorpora uma linguagem de marcação ou estruturação para documentos, a aplicação XML [36]. Uma vez criados os documentos de cada projeto, esses documentos devem ser armazenados em um banco de dados XML, favorecendo as consultas dos mesmos, com o propósito de buscar os DRs inicialmente armazenados na DocRationale.

Utilizando *tags* XML é possível desenvolver ferramentas de busca para melhorar a recuperação de informações para manutenção e reutilização dos documentos [16]. Haja vista todas essas características, optou-se por adotar os documentos XML para integrar os dados e os DRs a serem recuperados.

Para recuperar as informações de documentos XML, eles devem estar estruturados. O *XML Schema Definition* (XSD) [37] descreve a estrutura de um documento XML. Assim sendo, inicialmente, foi preciso definir o XSD. As páginas *web* dos projetos foram utilizadas para definir o XSD, de maneira semelhante ao processo descrito por Fumagalli et al. [14] para abstrair a estrutura de documentos.

Definida a estrutura do documento XML, o passo seguinte é criar os documentos. Para isso, algumas tecnologias como Cocoon [22] e XSP são utilizadas.

4.1 Cocoon

O Projeto Cocoon [22, 38] foi fundado em Janeiro de 1999 por Stefano Mazzocchi como um projeto de código aberto sob o *Apache Software Foundation*. É um *framework* de desenvolvimento *web* construído em torno dos conceitos de desenvolvimento *web* baseado em componentes. Esses conceitos são implementados no Cocoon em torno da noção de *pipelines* de componentes. Cada componente no *pipeline* se especializa em uma operação particular: um documento XML é passado através de um *pipeline*, que consiste em diversos passos de transformação do documento. Todo *pipeline* inicia com um *generator*, continua com ou sem *transformers*, e termina com um *serializer*. O *generator* é o ponto inicial para o *pipeline* e o responsável por entregar os eventos SAX no *pipeline*. O *generator* é construído para ser independente do conceito de arquivo. Se for possível gerar eventos SAX a partir de uma outra fonte, isso pode utilizado sem ter que ser via arquivo temporário. O *transformer* obtém um documento XML (ou eventos SAX) e o transforma em outro documento XML (ou eventos SAX). O *serializer* é responsável por

transformar eventos SAX em um formato de apresentação. Existem diversos s*erializers* que podem gerar diferentes formatos. A descrição de como o Cocoon é utilizado é descrito na próxima seção.

4.2 Criação dos Arquivos XML

Para cada projeto criado na DocRationale, deve existir um documento XML correspondente. As *tags* definidas no XSD devem ser inseridas nos documentos e os dados provenientes das consultas realizadas nas bases de dados e relacionados a cada uma das *tags*, devem ser também inseridos no documento XML. Deve-se ressaltar que a estrutura definida no XSD deve ser mantida, pois ela reflete a hierarquia das informações referentes aos DRs obtidos. Resumidamente, os documentos XML são gerados pelo Cocoon através do processamento de um documento XSP.

XSP (*Extensible Server Pages*) é uma linguagem de marcação dinâmica como JSP (*Java Server Pages*) ou ASP (*Active Server Pages*) que provê diretivas de código embutido. É fortemente integrada com o Cocoon.

Um documento XSP é um documento XML com *tags* especiais XSP, que pode ser utilizado como o ponto de partida de uma *pipeline* de processamento do Cocoon. O documento XSP é processado por um g*enerator* especial do Cocoon chamado *ServerPagesGenerator*, cujo processador converte o documento XSP em um programa fonte para uma particular linguagem de processamento. Através da execução do programa gerado, o documento XML resultante, descrito pelo documento XSP original, é gerado e intercalado com o conteúdo dinâmico gerado pelo código extraído a partir das diretivas de *tags* XSP. Em outras palavras, o documento XSP é utilizado pelo *ServerPagesGenerator* para criar um documento XML.

Neste trabalho, o documento XSP é o responsável por acessar as bases de dados da DocRationale, CoTeia e GroupNote e realizar as consultas necessárias para recuperar os dados corretos, seguindo a estrutura definida para o documento XML. O Cocoon processa esse documento XSP e gera os documentos XML correspondentes aos projetos existentes na ferramenta, que são armazenados no banco de dados eXist [27].

4.3 eXist

eXist [27] é uma base de dados XML de código aberto. O armazenamento de dados é baseado em árvores B+ e arquivos paginados. Os documentos são gerenciados em coleções hierárquicas, similar a arquivos armazenados em um diretório. eXist tem sua própria *Query Engine: XQuery*. Com extensões para dar suporte à pesquisa de documentos completos, as consultas podem transpor qualquer combinação possível de coleções ou documentos. Assim sendo, o banco de dados eXist possibilita definir diversos tipos de consultas que facilitam a busca dos DRs transcritos no documento XML de cada projeto.

4.4 Recuperação de DR dos Documentos XML

Como estratégia de recuperação, adotou-se a forma de pesquisa mais comum: busca por palavra-chave. Para a busca, pode-se selecionar um ou mais projetos a serem pesquisados. Também é possível direcionar a busca através da escolha de outras opções como fases e atividades dos projetos, artefatos, pessoas. A seleção das informações, inclusive a palavra-chave, para a busca e recuperação dos DRs dos projetos é realizada a partir da DocRationale. Assim, proporciona-se que os interesses dos desenvolvedores sejam declarados.

Como a ferramenta está implementada em PHP, a busca pelos DRs dos artefatos de software é realizada via uma classe PHP (PHP API) que possibilita acessar o eXist e realizar as consultas.

O resultado das pesquisas é exibido na DocRationale como uma lista de *links* para os DRs recuperados. Prevê-se que uma classificação desses *links* pelos diversos atributos que foram estabelecidos no acionamento da busca possam auxiliar na interpretação e visualização dos DRs a serem explorados.

Além disso, possíveis arquivos anexos relacionados aos artefatos também devem ser considerados na busca pelos DRs. Para facilitar a varredura desses arquivos, que podem ser de diversos formatos (Microsoft Word, PDF, TXT, RTF), eles são convertidos em HTML através de ferramentas de código aberto: wvWare, pdftohtml, txt2html e rtf-converter.

Os arquivos em formato texto devem ser convertidos assim que seu *upload* é realizado na DocRationale. As ferramentas utilizadas diferem para cada tipo de arquivo.

- **wvWare** [21] é uma aplicação, com uma série de opções de linhas de comando, provida com a biblioteca wv que permite acesso a arquivos Microsoft Word, com o propósito de convertê-los em outros formatos. Dentre os diversos formatos para os quais os arquivos Word podem ser convertidos estão: HTML 4.0, Latex, dvi, PS, PDF, TXT, WML, RTF.
- **pdftohtml** [20] é uma ferramenta que converte arquivos do formato *Portable Document Format* (PDF) nos formatos HTML e XML. É baseada no XPDF, um visualizador de código aberto para arquivos pdf.
- **txt2html** [2] é uma ferramenta que converte arquivos em formato de texto para arquivos HTML. Ela oferece suporte para cabeçalhos, listas, tabelas, caracteres de marcação simples e *hyperlinks*.
- **rtf-converter** [12] é uma aplicação em linha de comando para converter arquivos de formato *Rich Text Format* (RTF) em HTML.

A varredura desses documentos, para busca por palavras-chave, será realizada por alguma ferramenta, já existente, que realize esse tipo de pesquisa.

Com esse mecanismo, graficamente esquematizado na Figura 3, busca-se recuperar os DRs obtidos de maneira mais eficiente e ágil. No entanto, deve-se ressaltar que esse mecanismo de recuperação não poupa o desenvolvedor de ter que ler as informações recuperadas para encontrar, ou não, a informação que realmente deseja. Acreditamos que a exploração a partir dessas informações pode contribuir para que o desenvolvedor tome conhecimento de outras informações também relacionadas, e para as quais não teria condições de ao simplesmente navegar, procurar por elas.

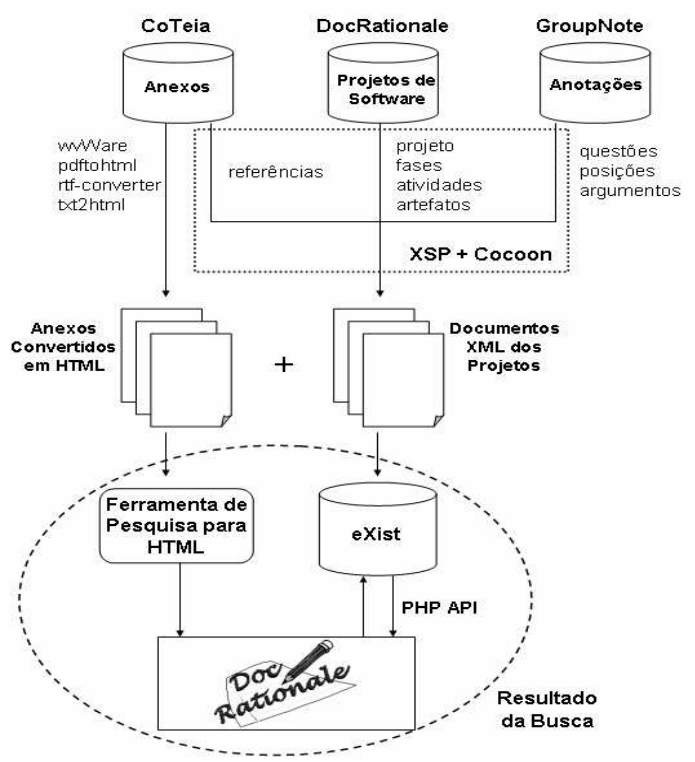


Figura 3: Mecanismo de recuperação de DR.

5. TRABALHOS RELACIONADOS

Uma abordagem baseada em XML para a verificação de documentos de requisitos de software é apresentada em [11]. Os requisitos de software são apresentados em XML e a linguagem XSLT é utilizada para verificar propriedades de qualidade desejadas, e para computar algumas métricas. O objetivo, nesse caso, é aplicar tecnologias XML, durante o processo de documentação de software, para verificar a qualidade da produtividade dos documentos de software e não são claramente mencionados os possíveis mecanismos para recuperação de informações relativas à DR e os próprios conteúdos dos documentos de requisitos.

Já a abordagem proposta em [4, 10] é a de mudar a representação de um código fonte para facilitar o uso de ferramentas e métodos mais poderosos de gerenciamento de documentos. É descrita uma aplicação XML que é utilizada para adicionar informação estrutural a arquivos não estruturados de código fonte. O texto pode, então, ser mais facilmente buscado, percorrido e transformado com o auxílio das *tags*. Assim, essa abordagem apresenta um conjunto mais completo de suporte à exploração de informações relacionadas com código-fonte de software. No entanto, o tratamento proposto se restringe a código-fonte, no caso, Java.

XML-GL [9] é uma linguagem gráfica de pesquisa para documentos XML. Um formalismo visual é utilizado para representar o conteúdo de documentos XML e a sintaxe para as pesquisas, permitindo que as mesmas sejam bastante complexas. Observa-se, no entanto, que a representação em XML-GL requer uma nova semântica das informações contidas nos documentos. No caso de informações referentes à documentação e ao DR de projetos de software, essa representação não é trivial.

6. CONCLUSÕES

As razões de projeto *(Design Rationale - DR)* constituem uma base de informações relacionadas às decisões tomadas e também às razões que propiciaram cada decisão, incluindo suas justificativas, alternativas consideradas, assim como o raciocínio empregado no desenvolvimento de um projeto. Nos projetos de software, a atividade de documentação é responsável pelo registro dos artefatos (ou seja, quaisquer tipos de documentos produzidos durante a produção do software) que são desenvolvidos. Assim, um documento XML se apresenta como mecanismo apropriado para apoiar essa atividade, propiciando que ligações sejam inseridas aos artefatos, para associar o DR correspondente.

Para possibilitar a exploração do DR registrado em documento XML, as estratégias de recuperação (navegação simples, navegação adaptativa, pesquisa e abordagem híbrida) foram analisadas.
Neste artigo foi apresentado um mecanismo para a busca mais rápida e objetiva do DR obtido e armazenado na ferramenta DocRationale, considerando o interesse dos desenvolvedores. Para isso são utilizadas tecnologias XML. Para recuperar o DR obtido, foi proposto um mecanismo, que tem como principal objetivo integrar informações sobre projetos de software, que estão distribuídas em diferentes bases de dados, em documentos XML, utilizando-se para isso tecnologias XML como o Cocoon e XSP. Esses documentos são armazenados em uma base de dados XML: eXist. A busca pelo DR armazenado é então possível a partir de consultas no eXist e como resultado, uma lista de *links* para as informações encontradas é exibida.
Embora o desenvolvedor não seja poupado de ter que navegar pela lista de *links* para encontrar a informação que realmente deseja, busca-se, com a definição desse mecanismo, estruturar as informações em um documento e tornar a recuperação de DR mais objetiva e rápida.
Como continuidade deste trabalho, diferentes alternativas de classificação dos *links* para os DR´s obtidos da busca serão implementadas, possibilitando que o desenvolvedor visualize os contextos dos DR´s, como por exemplo, quais os que ocorreram em determinado período de tempo, ou que envolveram um determinado membro da equipe de desenvolvimento. Assim, conforme interesse do desenvolvedor, o mecanismo de busca poderá auxiliar na exploração dos conteúdos de informação dos DRs obtidos. Finalmente, estudos experimentais serão realizados com o objetivo de avaliar com dados reais o mecanismo proposto.

References

[1] Adelson, B. and Soloway E. The Role of Domain Experience in Software Domain. *IEEE Transactions on Software Engineering*. Vol. 11, (1995).

[2] Andersen K. txt2html. http:// txt2html.sourceforge.net [February, 2004].

[3] Arruda Jr, C. R. E., Izeki, C. A. and Pimentel, M. G. C. CoTeia: Uma Ferramenta Colaborativa de Edição Baseada na Web. Anais doWorkshop de Ferramentas e Aplicações de VII SBMIDIA, (2002).

[4] Badros, G. J. Javaml: A Markup Language for Java Source Code. *Proceedings of the International World Wide Web Conference - WWW9*, (2000).

[5] Borges, L. S. and Falbo, R. A. Managing Software Process Knowledge. *Proceedings of the International Conference on Computer Science, Software Engineering, Information Technology, e-Business and Applications – CSITea'2002*, (2002), pp. 227-232.

[6] Brin, S. and Page, L. The Anatomy of Large-Scale Hypertextual Web Search Engine. http://www7.scu.edu.au/programme/fullpapers/1921/com1921.htm [February, 2003], (2001).

[7] Brusilovsky P. Methods and Techniques of Adaptive Hypermedia. *User Modeling and User Adapted Interaction*. Vol. 6, No. 2-3, (1996), pp. 87-129.

[8] Brusilovsky P. Adaptive Hypermedia. *User Modeling and User Adapted Interaction*. Vol. 11, No. 1, (2001), pp. 87-110.

[9] Ceri, S., Damiani, E., Fraternali, P., Paraboschi S. and Tanca L. XML-GL: A Graphical Language for Querying and Restructuring XML Documents. *Proceedings of the 8th International World Wide Web Conference - WWW8*, (1999).

[10] Collard M., Maletic, J. I. and Marcus, A. Supporting Document and Data Views of Source Code. *Proceedings of the ACM Symposium on Document Engineering – DocEng'02*, (2002), pp.34-41.

[11] Duran, A., Ruiz, A., Bernadez, B. and Toro, M. Verifying Software Requirements with XSLT. *ACM Software Engineering Notes*. Vol. 27, (2003).

[12] Ford, G. rtf-converter. http://freshmeat.net/projects/rtfconverter [February, 2001], (2003).

[13] Francisco S. D., Izeki, C. A., Paiva, D. M. B. and Fortes, R. P. M. Um Sistema de Apoio à Utilização de Design Rationale de Artefatos de Software. Anais da XXIX Conferência Latinoamericana de Informática – Clei 2003, (2003).

[14] Fumagalli, L. C., Pansanato, L. T. E amd Fortes, R. P. M Documentation Process in Interactive Systems – A Case Study to Abstract its Structure. *Proceedings of the 2nd International Information ans Telecommunication Technologies Symposium – I2TS 2003*, (2003).

[15] Gruber, T. R. and Russel, D. M. Design Knowledge and Design Rationale: A Framework for Representation, Capture and Use. *Technical Report KSL 90-45 – Knowledge Systems Laboratory*, (1991).

[16] Harold, E. R. and Means, W. S. *XML in a Nutshell*, O'Reilly. 2002.

[17] Hu, W. C., Chen Y., Schmalz M. S. and Ritter, G. X. An Overview of World Wide Web Search Technologies. *Proceedings of the 5th World Multi-Conference on System, Cybernetics and Informatics – SCI2001*,(2001).

[18] Izeki, C. A., Arruda Jr, C. R. E and Pimentel, M. G. C. An XML-based Infrastructure Supporting Collaborative Annotations as First-class Hyperdocuments. *Proceedings of the VII Brazilian Symposium of Multimedia and Hypermedia Systems*, (2001), pp. 173-186.

[19] Kobayashi M. and Takeda, K. Information Retrieval on the Web. *ACM Computing Surveys*. Vol.32, No. 2, (2002).

[20] Kruk, M. pdftphtml. http://pdftphtml.sourceforge.net [February, 2004], (2003).

[21] Lachowics, D. wvWare. http://wvware.sourceforge.net [February, 2003], (2000).

[22] Langham, M. and Ziegeler C. *Cocoon: Building XML Applications*. New Riders Publishing, 2002.

[23] Lee, J. Design Rationale Systems: Understanding the Issues. *IEEE Expert/Intelligent Systems and Their Applications*. Vol. 12, No. 3, (1997), pp.78-85.

[24] Lindvall M. and Sandahi, K. How well do Experienced Software Developers Predict Software Change? *The Journal of Systems and Software*. No. 43, (1998), pp.19-27.

[25] Markkula, M. Knowledge Management in Software Engineering Projects. *Proceedings of the 11th International Confrence on Software Engineering and Knowledge Engineering – SEKE'99*, (1999).

[26] McCall, R. J. PHI: A Conceptual Foundation for Design Hypermedia. *Design Studies*. Vol. 12, No. 1, (1991), pp.30-41.

[27] Meier, W. eXist: An Open Source Native XML Database. *Revised Papers from the Web and Database-Related Workshops on Web, Web Services, and Database – NODe 2002*, (2002), pp. 169-183.

[28] Moran, T. P. and Carroll, J. M. *Design Rationale Concepts, Techniques and Use Computers, Cognition and Work*. New Jersey: Lawrence Erlbaum Associates, 1996.

[29] Nielsen, J. *Multimedia and Hypertext: The Internet and Beyond*. AP Professional, 1995.

[30] Pressman, R. S. *Software Engineering: A Practitioner's Approach*. Fifth Edition, McGraw Hill, 2000.

[31] Regli, W. C., Hu, X., Atwood, M. and Sun, W. Survey of Design Rationale Systems: Approaches, Representation, Capture and Retrieval. *Engineering with Computers: An International Journal for Simulation-Based Engineering* (Special issue on Computer Aided Engineering in honor of Professor Steven J. Fenves). Vol. 16, (2000), pp. 209-235.

[32] Shipman F, and McCall, R. Integrating Different Perspectives on Design Rationale: Supporting the Emergence of Design Rationale from Design Communication. *Artificial Intelligence in Engineering Design, Analysis and Manufacturing*. Vol. 11, No. 2, (1997), pp. 141-154.

[33] Souza, C., Santos, D. B., Dias, K. L. and Wainer, J. A. Model and Tool for Semi-Automatic Recording of Design Rationale in Software Diagrams. *Proceedings of the String Processing and Information Retrieval Symposium and International Workshop on Groupware*, (1999), pp.306-313.

[34] Trevor, B., Weippl, E. and Winiwarter, W. A Modern Approach to Searching the World Wide Web: Ranking Pages by Inference over Content. *Proceedings of the 14th International Conference Applications of Prolog*, (1998).

[35] Wangenheim, C. G., Althoffl, K. D. and Barcia, R. M. Intelligent Retrieval of Software Engineering Experienceware. *Proceedings of the 11th International Conference on Software Engineering and Knowledge Engineering – SEKE'99*, (1999).

[36] World Wide Web Consortium. *Extensible Markup Language (XML) 1.0*. Second Edition, W3C Recommendation. http://www.w3.org/TR/2000/REC-xml-20001006 [May, 2004], (2000).

[37] World Wide Web Consortium. *XML Schema 1.1*. W3C Recommendation. http://www.w3.org/XML/Schema [May, 2004], (2001).

[38] The Apache Software Foundation. *Cocoon 2.1*. Apache Cocoon Project. http://cocoon.apache.org/2.1 [april, 2004], (2004).

Infraestructura de Realidad Virtual Multiplataforma

Daniel Mejia
Universidad de los Andes, Ingeniería de Sistemas y Computación,
Bogotá, Colombia
dan-meji@uniandes.edu.co

and

Pablo A. Figueroa
Universidad de los Andes, Ingeniería de Sistemas y Computación,
Bogotá, Colombia
pfiguero@uniandes.edu.co

and

J. T. Hernández
Universidad de los Andes, Ingeniería de Sistemas y Computación,
Bogotá, Colombia
jhernand@uniandes.edu.co

Abstract

We present a software infrastructure for the development of multiplatform, virtual reality applications. We use open–source toolkits and the previously published InTml architectural framework in order to provide an environment where developers can modify in a predefined way devices, interaction techniques, and content quality. Our main contributions are the uniform execution environment for portable virtual reality applications over a multi–framework setup, the separation of responsibilities during development, and the analysis of important variation points in such type of applications. We show two simple applications as a proof of concept of this infrastructure

Keywords: Virtual Reality, Open–Source, InTml, Development Environments for VR, Portable VR Applications.

Resumen

Presentamos una infraestructura software para el desarrollo de aplicaciones de realidad virtual multiplataforma. Usando herramientas de distribución libre y la arquitectura de software InTml propuesta previamente, permitimos al desarrollador crear aplicaciones en las que hay una forma planeada de cómo variar los dispositivos de entrada–salida, las técnicas de interacción utilizadas, y la calidad del contenido. Las contribuciones más importantes de este trabajo son la definición de un ambiente de ejecución uniforme para aplicaciones de realidad virtual portables, la división de responsabilidades en el desarrollo y el análisis de los cambios importantes en aplicaciones de este tipo. Al final mostramos dos aplicaciones que hemos desarrollado bajo esta plataforma

Palabras claves: Realidad virtual, Software libre, InTml, Ambientes de desarrollo para RV, aplicaciones de RV portables

1 INTRODUCCIÓN

Una aplicación de realidad virtual (RV) permite a su usuario interactuar con información mediante novedosos dispositivos y técnicas de interacción, generalmente en un espacio tridimensional (3D). A pesar de un desarrollo de más de treinta años, son pocas las aplicaciones exitosas, principalmente por las expectativas tan altas que se tenían al principio y el consecuente desinterés. Sin embargo, aplicaciones en áreas como

petróleos, diseño de automóviles y entretenimiento han demostrado lo importante de esta tecnología y sus ventajas. Una aplicación de RV típica combina eventos de varios dispositivos de entrada no convencionales como trackers y cámaras, cambia el estado de un modelo con representación 3D y lo muestra mediante imágenes o sonido tridimensional. El desarrollo de este tipo de aplicaciones puede facilitarse actualmente con el uso de algún ambiente de desarrollo, bien sea comercial o académico. Sin embargo, dichos ambientes son aún complicados de usar, están dirigidos a cierto tipo de ambiente computacional o dispositivos de entrada salida y limitan el tipo de cambios en el software, necesarios para probar distintas opciones en un análisis de factibilidad de uso.

Este trabajo muestra nuestro análisis de las variaciones importantes en una aplicación de RV, y una infraestructura que soporta estas variaciones. El análisis de variaciones destaca qué elementos deben ser fácilmente cambiables en este tipo de aplicaciones, para asegurar que el desarrollador pueda probar diferentes alternativas de hardware y software y que la aplicación puede evolucionar organizadamente.

El artículo está dividido en las siguientes partes: descripción del trabajo previo, requerimientos, infraestructura, ejemplos y por último conclusiones y trabajo futuro.

2 TRABAJOS PREVIOS

Existen varios ambientes de desarrollo para aplicaciones de RV, con diversas capacidades y limitaciones. Muchos de ellos se valen de archivos de configuración, en los cuales se pueden dar valores a parámetros que pueden variar de una instalación a otra. Algunos cuentan con herramientas para editar dichos archivos [3], mientras que en otros debe usarse un simple editor de texto [14]. La tabla 1 describe las principales características de diversas alternativas.

Toolkit	Capacidades principales
MRToolkit [14]	Abstracción dispositivos Archivos con parámetros
CAVELib [18]	Abstracción dispositivos Distribución
Performer [13]	Grafo para la escena Capacidades computacionales
WTK [12]	Grafo para la escena Abstracción dispositivos Ambiente de desarrollo rápido
Alice [17]	Ambiente de desarrollo rápido
VRJuggler [3]	Abstracción dispositivos Editor de archivos de configuración Abstracción de librería gráfica Integración con otras herramientas
X3D [19]	Grafo para la escena Abstracción del ambiente de ejecución
VRPN [11]	Abstracción dispositivos Distribución
InTml [5]	Arquitectura de software para RV Integración con otras herramientas

Table 1: Características principales en algunas herramientas de desarrollo para aplicaciones de RV.

Las fortalezas más comunes en herramientas para desarrollo de RV están relacionadas con la abstracción en software de dispositivos de entrada y la organización de los elementos gráficos mediante un grafo. Sin embargo, una aplicación de RV completa consta de otros elementos, tales como técnicas de interacción, que no están directamente representados en elementos de los sistemas existentes [1]

3 VARIABILIDAD EN APLICACIONES DE RV

Nuestro laboratorio ha desarrollado varias aplicaciones de RV en el pasado, que han demostrado ciertos conceptos en el área, pero que también nos han mostrado limitaciones de la tecnología y de las herramientas

[1] Java3D [8] define el comportamiento dentro del mundo virtual como un elemento principal, pero no existen aún muchos ejemplos de dichos elementos y su integración con el resto del framework.

de base que hemos utilizado. Fruto de esta experiencia previa hemos identificado el siguiente conjunto de requerimientos, que deseamos satisfacer en un nuevo ambiente de desarrollo:

- El conjunto de dispositivos de entrada y de salida de una aplicación debe ser configurable. Debe ser posible utilizar dispositivos remotos, simular un dispositivo mediante otros, o cambiar totalmente la interfaz hardware de una aplicación.
- Se debe permitir el cambio de técnicas de interacción, como un mecanismo para probar diversas alternativas en el diálogo con el usuario y como forma de adaptar de manera óptima nuevos dispositivos de interacción.
- Debe poderse elegir la calidad de las gráficas mostradas dependiendo de las capacidades computacionales de la máquina donde corra la aplicación.
- Se debe poder contar con una librería de acceso a diversos dispositivos de entrada salida, diversos tipos de contenido y diversas técnicas de interacción.

La infraestructura que planteamos a continuación trata de satisfacer dichos requerimientos, mediante la integración de herramientas previamente nombradas.

4 INFRAESTRUCTURA

Nuestra infraestructura está compuesta del conjunto de dispositivos disponibles en nuestro laboratorio de informática gráfica, la plataforma de software disponible en éstos equipos y los procedimientos diseñados para el desarrollo de aplicaciones en dicha plataforma. A continuación se analizan estos elementos en detalle.

4.1 Hardware

Actualmente el laboratorio cuenta con diversos dispositivos que combinados en una aplicación pueden generar varios ambientes de RV. Estos dispositivos varían desde el soporte más básico en un PC convencional, hasta los necesarios dentro de un ambiente de proyección.

Un primer ambiente de RV básico es logrado a través de un PC convencional, en donde la interacción se realiza con el mouse y el teclado, siendo útil para aplicaciones de visualización y rendering, como por ejemplo procesamiento de imágenes.

El segundo ambiente es un "fish tank desktop" [2] con visualización estéreo, logrado por medio de un PC con buenas capacidades gráficas (tarjeta Quadro FX 3000 [9] y 2GB en RAM), un monitor de alto refresco vertical (120Hz) y trackers Flock of Birds para la interacción [16]. Este ambiente ha sido utilizado para la reconstrucción de imágenes médicas. El tercer ambiente disponible se basa en un PC convencional que despliegua la imagen en un i-glasses HMD (Head Mounted Display) de baja resolución.

Estos ambientes pueden utilizar diversos dispositivos de interacción, disponibles en otra máquina de manera remota, como guantes (izquierdo y derecho) [15], joystick con retroalimentación de fuerza [4] y un gamepad [7].

Finalmente el laboratorio se encuentra en este momento en proceso de adquisición de un ambiente proyectivo que le permita generar un sistema CAVE, para lograr así una inmersión completa del usuario en mundos generados.

4.2 Software

Nuestro sistema tiene como núcleo de integración VrJuggler ya que este ofrece interacción con otro tipo de tecnologías como Performer o VTK [6] y adicionalmente permite interactuar con diversidad de dispositivos.

Sobre esta infraestructura de software corre un framework InTml [5] adaptado para este ambiente. Este framework consta de un sistema configurable, que permite acceder a los distintos recursos con archivos de configuración, permitiendo así que las aplicaciones puedan cambiar el ambiente de RV de acuerdo a las necesidades. Para el desarrollador que usa este framework, una aplicación consiste en un conjunto de filtros (componentes) que representan dispositivos, técnicas de interacción y contenido; y que pueden conectarse entre sí en el desarrollo de una aplicación.

Adicionalmente a esto la infraestructura esta montado sobre una red TCP/IP para poder independizar los dispositivos del ambiente de desarrollo. Los dispositivos se encuentran configurados en un arquitectura cliente - servidor para lo cual utilizamos VRPN como servidor de dispositivos. Mediante esta librería es posible acceder a los trackers, joystick, gamepad y demás dispositivos desde cualquier máquina de nuestro laboratorio.

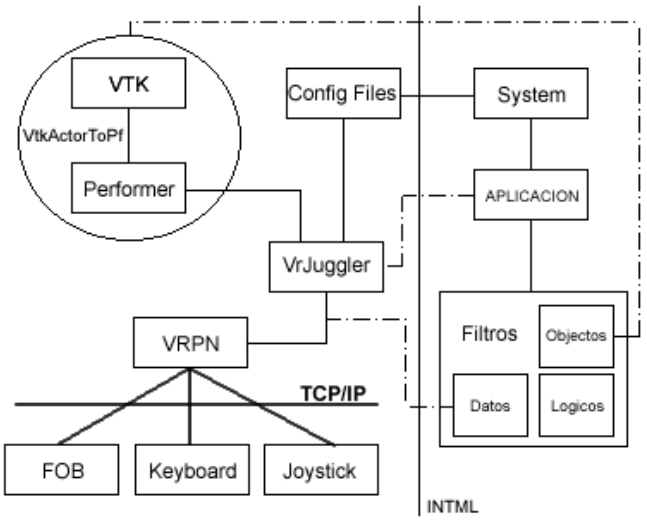


Figure 1: Descripción de la arquitectura de software

La integración de VRPN, VRJuggler e InTml crea una estructura particular de software, debido a que se debe modularizar la aplicación para correr sobre un sistema InTml. Estos se hace con la creación de filtros que nos permiten interactuar con todas las tecnologías. El diseño de toda la arquitectura de software utilizada se puede ver en el grafica anterior(imágen. 1), donde se muestra la división entre la plataforma e Intml, así como el acceso a cada una de las estas, representada por las lineas punteadas que salen de los filtros.

Estos filtros necesitan obtener datos de los dispositivos y acceder a objetos Performer. Para la obtención de datos aprovechamos la facilidad de VrJuggler para interactuar con VRPN por medio de sus archivos de configuración; y para acceder a los objetos lo que se hace es obtener un nodo Performer que contenga objetos y transformaciones. En el caso de un objeto VTK la transformación a un objeto Performer se realiza por medio de VTKActorToPf [10].

El desarrollo actual se centra en dos partes, la primera construir filtros InTml y la segunda usar estos filtros para desarrollar nuevas aplicaciones. Una vez se tiene los filtros, la creación de la aplicación se reduce a la unión de los mismos, simplificando así el desarrollo, ya que crear la aplicación se reduce a unir los filtros que se han desarrollado sin profundizar en el detalle técnico de cada uno de ellos. La creación es sencilla: los dispositivo se conectan a las técnicas de interacción y éstos a su vez se conectan con los objetos para modificarlos. Más adelante se mostrarán ejemplos de aplicaciones que corren bajo este esquema, donde se ilustrará claramente la unión de los filtros.

Por último, cabe destacar la división de trabajo que este ambiente de desarrollo permite: Hay un trabajo inicial de los desarrolladores para poner a punto un conjunto de filtros en InTml, los cuales pueden ser reutilizados más adelante en otras aplicaciones. De esta manera, el desarrollador final no tiene que entrar en todos los detalles intrínsicos en una aplicacion de RV, ya que éstos se encapsulan dentro de los filtros.

4.3 Ambiente Integrador

Así como la creación de la aplicación es sencilla, también se desea que esta sea flexible en su configuración de acuerdo a las necesidades del usuario final. Actualmente se han seleccionado ciertas variables de configuración que se consideran relevantes para el usuario y estas se pueden modificar por medio de parámetros en línea de comandos.

Se esta trabajando para que la sistema soporte no solo líneas de comando sino archivos de configuración ant [1], por medio de los cuales el usuario puede modificar las variables relevantes de una manera más estructurada. Los parámetros que actualmente consideramos son:

- Simulado (true o false). Permite cambiar los dispositivos que la aplicación usa por un ambiente simulado en el desktop. Se usa para pruebas principalmente
- 3d (true o false). Cambia la visualización de la aplicación a modo 3D o 2D. Util principalmente en el ambiente "fish tank".

- Calidad imagen (baja, media, alta). Permite variar la calidad de la producción de la imagen, para acelerar su despliegue [2].

4.4 Procedimientos

El primer proceso importante en nuestro laboratorio es el proceso de calibración de dispositivos, especialmente para los trackers. Este proceso se desarrolla antes de la ejecución de una aplicación y periódicamente para corregir cambios en el ambiente. En el caso de los trackers, se construyo una mesa con marcas de calibración cada 10cm, y que puede variar de altura, para tomar datos del tracker en una grilla 3D de 70cm de lado. Estos datos se pasan a un archivo de configuración que es leido por VRJuggler para calibrar los datos que arroja el dispositivo [3]

A un nivel de software se puede modificar los dispositivos de entrada y salida los cuales están representados como filtros. Hacer un cambio de dispositivos es hacer un cambio de filtros, buscar el filtro del dispositivo que se quiere, utilizarlo y conectarlo de una manera adecuada a los demás filtros en la aplicación. Al igual que con los dispositivos hacer un cambio en las técnicas de interacción se logra con un cambio de filtro. Entre mayor sea la cantidad de filtros desarrollados es mayor la cantidad de elementos disponibles para el desarrollo de aplicaciones.

Como ya vimos, cambios de rendimiento y opciones de alto nivel son seleccionadas mediante parámetros en la línea de comandos, dentro de los que se encuentra la calidad de la imagen y la visualización en estéreo, logrando así satisfacer los requerimientos de variabilidad planteados. Los parámetros a alto nivel se reflejan en el tratamiento interno de los filtros que representan objetos gráficos. En la implementación actual de filtros que representan objetos se maneja de dos maneras completamente distintas: en Performer se mantiene un estándar de nombres y en VTK se mantiene un parámetro dentro del filtro que se crea al empezar la aplicación.

El proceso de configuración del hardware es un proceso de creación de archivos en las diferentes tecnologías, principalmente en VRPN, donde se registra el dispositivo, dándole las propiedades necesarias. Posteriormente se crean los archivos VrJuggler necesarios para leer los dispositivos desde VRPN, dandoles un nombre significativo que les permita ser registrados en el sistema. Por último se configura un filtro en InTml, que representa el dispositivo y con esto crear los archivos necesarios para cada uno de los ambientes con los que se quiere trabajar.

Para crear un nuevo filtro se debe crear una clase en C++, que hereda de una clase genérica, de acuerdo a lo que se desea (un objeto, un dispositivo o un filtro). La nueva clase representa el comportamiento esperado del nuevo filtro, junto con las salidas y entradas esperadas.

Por último, consideramos un procedimiento para la ejecución de la aplicación. Actualmente se corre como una aplicación convencional que recibe los parámetros por línea de comandos, estos modifican la aplicación de acuerdo a sus valores. La idea final es correr la aplicación por medio de ant, lo que dividiría el proceso en dos, el primero un pequeño programa que lee parámetros en línea y cambia el archivo de propiedades, después se correría ant con el archivo creado.

5 APLICACIONES DE EJEMPLO

5.1 Un Visualizador Genérico

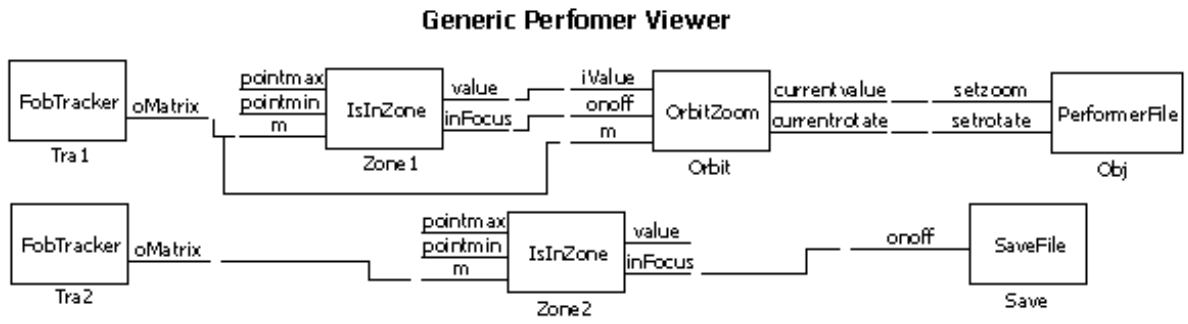


Figure 2: Visualizador Genérico

[2] Actualmente implementado para objetos gráficos de VTK, y en desarrollo para objetos de Performer.
[3] Estamos trabajando en un proceso que utiliza únicamente VRPN, para utilizar los datos de calibración a un nivel más bajo.

El objetivo de esta aplicación es dar las opciones típicas de un visualizador (zoom y rotación), para cualquier tipo de objeto Performer, la gráfica anterior (imágen. 2) muestra el diseño de la aplicación basándose en filtros, a continuación se describen cada uno de estos.

- Fobtracker: representa un tracker Flock of Birds, el cual retorna la matriz de transformación del punto, representada como un matrix44f de gmtl (librería de conceptos matemáticos, parte de VRJuggler).
- IsInZone: filtro de interacción que calcula si el punto que llega al puerto m, se encuentra dentro de la región delimitada entre pointmin y pointmax, retornando un valor verdadero (inFocus) y la correspondiente posición normalizada entre 0 y 1 en el espacio (value).
- OrbitZoom: filtro de interacción que calcula el zoom y la rotación . El valor de la translación con base en el valor recibido en iValue y la rotación con base en la matriz, estos valores son calculados si se encuentra activado el filtro (onoff) y envia los correspondientes valores a los salidas currentvalue y currentrotate.
- PerformerFile: Carga un archivo de un modelo Performer y crea el nodo correspondiente de translación y rotación, adicionalmente ofrece cambiar estos valores por medio de setrotate y setzoom.
- SaveFile: Al estar activo guarda el estado de las variables del usuario en un archivo, para poder recuperar posteriormente el mismo estado.

La aplicación une estos filtros, y obtiene el visualizador, donde se interactúa con un tracker, el primero, para zoom - rotate y el segundo para salvar el estado. Desde el punto de vista de ejecución, un usuario debe correr el programa mediante los parámetros previamente nombrados, mas el directorio donde se encuentran los archivos del objeto Performer.

5.2 Un Visualizador de Imágenes de Scanner

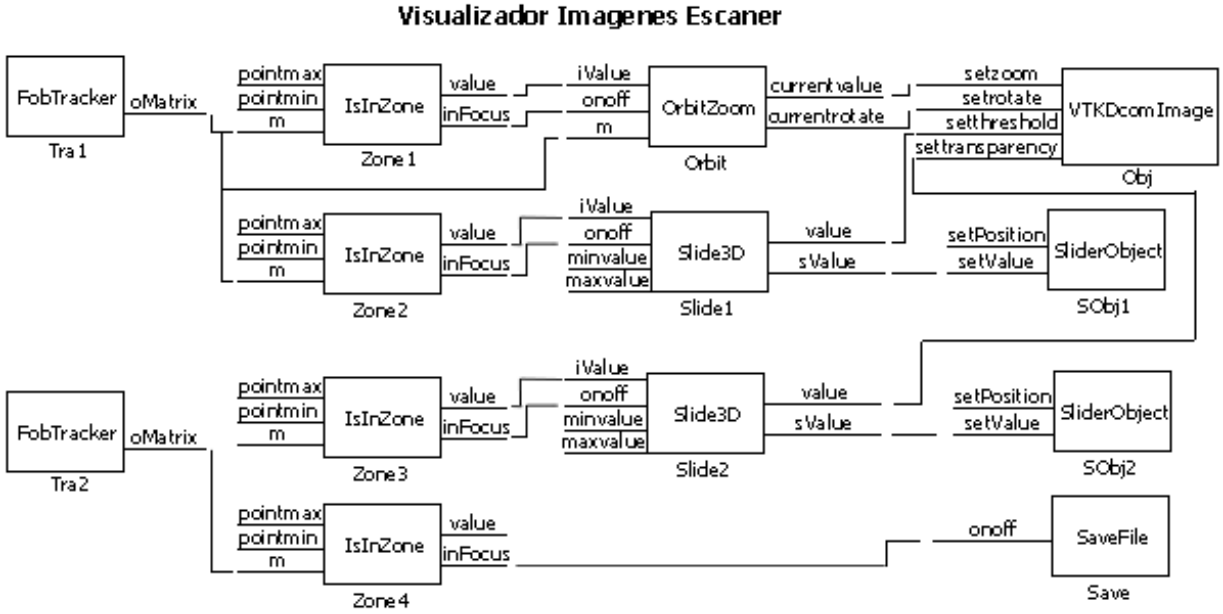


Figure 3: Visualizador de imgenes Escaner

Esta aplicación es menos genérica que la anterior, pero permite más opciones, ya que no solo es un visualizador sino que permite modificar los parámetros de reconstrucción. Esta aplicación utiliza muchos de los filtros de la anterior aplicación (imágen. 3) y otros nuevos que se especificarán a continuación.

- Slide3D: Se encarga de la lógica de un slider, calculando su posición y enviándola de dos maneras, la primera constantemente y la segunda al salir del slider (value, sValue).
- SliderObject: Es la forma de un slider generado en VTK (vtkCubeSource), y transformado a Performer por vtkActorToPf.

- VTKDComImage: Este filtro reconstruye dos superficies de acuerdo a dos umbrales establecidos, una de estas es fija y la otra con posibilidad de cambiar su umbral de reconstrucción y su transparencia. Toda la reconstrucción se hace por medio de VTK y su respectiva transformación en vtkActorToPf.

6 CONCLUSIONES Y TRABAJO FUTURO

Con toda esta infraestructura se logra obtener un banco de trabajo muy completo, que permite el desarrollo de aplicaciones de RV muy fácilmente, ya que se centra en el diseño de la aplicación y no en los aspectos técnicos que hay por debajo (división de responsabilidades) y además permite que la reutilización de las aplicaciones de RV sea confiable y efectiva.

La facilidad en los cambios de dispositivos, técnicas de interacción y calidad de los modelos permite no solo que una aplicación de RV, sea transportable a cualquiera de nuestros ambientes de RV, sino que además le da la posibilidad al desarrollador de probar diferentes configuraciones en el momento del desarrollo.

El sistema montado es efectivo, pero al ser basado en los filtros son muy pocas las aplicaciones que se pueden lograr, dado la poca cantidad de ellos actualmente. Pensamos desarrollar nuevos filtros con técnicas de interacción, dispositivos, y objetos, que nos permitan contar con una librería más completa para el desarrollo de aplicaciones.

Estamos trabajando también en el desarrollo de un "browser de InTml", aplicación que nos permita leer el archivo fuente de InTml (escrito en XML) y cargar dinámicamente el código de los filtros. Esta utilidad ya existia en un prototipo anterior en Java.

Por último, deseamos contar con una aplicación grafica que permita crear una nueva aplicación InTml en una forma más amigable para el usuario final, por medio de un lenguaje visual de programación.

References

[1] Apache. The apache ant project. http://ant.apache.org/, 2004.

[2] Kevin W. Arthur, Kellogg S. Booth, and Colin Ware. Evaluating 3d task performance for fish tank virtual worlds. *ACM Trans. Inf. Syst.*, 11(3):239–265, 1993.

[3] Allen Bierbaum, Christopher Just, Patrick Hartling, Kevin Meinert, Albert Baker, and Carolina Cruz-Neira. VR Juggler: A Virtual Platform for Virtual Reality Application Development. In *Proceedings of IEEE Virtual Reality*, pages 89–96, 2001.

[4] Microsoft Co. Sidewinder force feedback. http://www.microsoft.com/hardware/sidewinder/FFB2.asp, 2003.

[5] Pablo Figueroa, Mark Green, and H. James Hoover. InTml: A Description Language for VR Applications. In *Web3D 2002 Symposium Proceedings*, pages 53–58, 2002.

[6] Kitware. The visualization toolkit. http://public.kitware.com/VTK/, 2004.

[7] Logitech. Logitech dual action gamepad. http://www.logitech.com/index.cfm/products/details/US/EN,CRID=11,C
2004.

[8] Sun Microsystems. Java 3D Home Page. http://java.sun.com/products/ java-media/3D/index.html, 1997.

[9] NVIDIA. Nvidia quadro fx. http://www.nvidia.com/page/quadrofx_family.html, 2004.

[10] Paul Rajlich. vtkactortopf. http://brighton.ncsa.uiuc.edu/ prajlich/vtkActorToPF/, 2004.

[11] ll Russell M. Taylor, Thomas C. Hudson, Adam Seeger, Hans Weber, Jeffrey Juliano, and Aron T. Helser. VRPN: A device-independent, network-transparent VR peripheral system. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 55–61. ACM Press, 2001.

[12] Sense8. Virtual reality development tools. The sense8 product line. http://www.sense8.com/products/index.html, 2000.

[13] SGI. Iris performer home page. http://www.sgi.com/software/performer, 2003.

[14] Chris Shaw, Jiandong Liang, Mark Green, and Yunqi Sun. The decoupled simulation model for virtual reality systems. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 321–328. ACM Press, 1992.

[15] 5DT Technologies. 5dt data glove 5. http://www.5dt.com/products/pdataglove5.html, 2002.

[16] Ascension Technologies. Flock of birds. http://www.ascension-tech.com/products/flockofbirds.php, 2004.

[17] Carnegie Mellon University and University of Virginia. Alice: Easy interactive 3D graphics. http://www.alice.org, 1999.

[18] VRCO. Cavelib library. http://www.vrco.com/products/cavelib/ cavelib.html, 2003.

[19] Web3D Consortium. Extensible 3D (X3D^{TM}) Graphics. Home Page. http://www.web3d.org/x3d.html, 2003.

El patrón multi-visualización para la generación de distintas presentaciones en un sistema de comercio electrónico

José R. Gulias, Victor M. Gulias, Alberto Valderruten, Carlos Abalde

MADS Group - LFCIA
Departamento de Computación, Universidad de La Coruña
Campus de Elviña, 15071 La Coruña, España
{gulias,valderruten,carlos}@lfcia.org

Resumen

El sistema de *comercio electrónico* SCED dispone de una gran variedad de opciones y facilidades, orientadas todas ellas a favorecer las compras de los usuarios. Al interactuar con SCED se deberían tener en cuenta las características y capacidades de cada dispositivo de cara a sacarle un rendimiento óptimo a cada uno de ellos. En este trabajo se comenta la solución empleada, presentada bajo la forma de patrón de diseño: El Patrón Multi-Visualización. Este se utiliza para estructurar todo el proceso de generación de las presentaciones para el usuario.

1 Introducción

Los patrones de diseño ([GHJV95], [Dou03]) son soluciones a problemas comunes, que han sido utilizadas con éxito en el pasado y que son suficientemente generales para ser reutilizadas ante otros problemas similares.

Por lo tanto, el uso de patrones permite que los desarrolladores de aplicaciones no tengan que plantearse nuevamente soluciones a problemas comunes cada vez que se encuentren ante la construcción de un nuevo sistema. La solución más adecuada ante la aparición de un problema conocido es aplicar un patrón que se sabe que es útil para la resolución de ese problema y adaptarlo a sus características particulares.

Todo ello implica una reducción en el esfuerzo y en el coste del desarrollo de un sistema, y una mayor claridad y reusabilidad del código generado, características todas ellas inherentes al uso de patrones en el desarrollo de *software*.

Y, dentro del terreno del desarrollo de aplicaciones, hay que destacar la importancia que han ido adquiriendo a lo largo de los últimos años un conjunto de sistemas agrupados bajo la categoría de aplicaciones de *comercio electrónico* ([Men98], [TS98]).

Actualmente se acostumbra a identificar el concepto de *comercio electrónico* con el comercio a través de Internet (*Internet Commerce*), aunque hay que subrayar que esta es sólo una de las modalidades posibles de *comercio electrónico*. En este artículo, nos centraremos en este término como intercambio de dinero a través de redes abiertas como Internet a cambio de bienes, información o servicios del algún tipo.

Por lo tanto, las aplicaciones de *comercio electrónico* serán aquellas que permitirán a los usuarios acceder desde sus propias casas a la información de todos aquellos productos que tiene a la venta una tienda y realizar sus compras sin necesidad de desplazamientos.

La importancia de estas aplicaciones y sus grandes expectativas de futuro, nos llevan a tener en consideración la relevancia que tiene facilitar a los posibles usuarios las compras en cuanto sea posible, y esto incluye el método de acceso que estén usando.

El desarrollo tecnológico ha desembocado en la aparición en el mercado de un gran número de dispositivos que se adaptan a las preferencias y limitaciones de los usuarios. Cada uno de estos dispositivos cuenta con unas características propias como, por ejemplo, el tamaño o el lenguaje que entiende, que hacen que una misma página no pueda ser utilizada en todos estos dispositivos.

Por lo tanto, parece necesario, dada la importancia que se espera que tengan este tipo de aplicaciones, construir sistemas en los cuales se ofrezca una visualización que se adapte a las particularidades de los distintos dispositivos y faciliten al máximo la interacción de los potenciales clientes.

2 Sistema de Comercio Electrónico SCED

SCED (Sistema de Comercio Electrónico Distribuido) ([Fer01]) es una aplicación cubre las funcionalidades básicas de venta y gestión de un conjunto de tiendas (*mall*). El sistema se centra en el escenario B2C (*Business to Consumer*) proporcionando las funciones requeridas por los distintos actores que interactúan con el sistema, tanto usuarios, potenciales clientes de la aplicación, como administradores (de tienda o de *mall*).

Además, este sistema cuenta con una serie de peculiaridades en lo que respecta a su arquitectura e implementación. Para la implementación de este prototipo se utilizó el lenguaje funcional distribuido Erlang ([ERL01], [AWWV95]), y se manejaron aplicaciones y patrones específicos de este lenguaje de programación.

Una de las características destacables que ofrece el prototipo implementado es la personalización del contenido de las páginas en función del perfil del usuario y del navegador que esté usando para acceder al sistema. Esto se consiguió mediante la puesta en marcha de dos subsistemas independientes: uno encargado de generar un documento XML ([XML01],[PM99]) a partir

de la información del usuario y del entorno, y otro responsable de seleccionar y aplicar una hoja de estilos XSL para obtener la visualización en el formato deseado en función del navegador usado.

SCED está diseñado para funcionar de manera distribuida sobre los nodos de un *cluster*, y cumple otra serie de características interesantes como la escalabilidad, tolerancia a fallos, concurrencia, ...

2.1 Actores del sistema

Se estableció una jeraquía de actores durante el diseño del sistema, pensando en los distintos roles que podrían interactuar con la aplicación. Cada uno de ellos, podrá realizar una serie de acciones determinadas sobre el sistema, a las que se añaden otras más específicas a medida que descendemos por la jerarquía.

1. Usuario: Este actor se corresponde con cualquier potencial visitante del *mall* de tiendas.

2. Usuario anónimo: Este actor representa a aquellos usuarios que no reconoce el sistema, bien porque sea su primera visita o bien porque aún no se haya registrado en el sistema y este sea incapaz de identificarlo correctamente.

3. Usuario registrado: Este actor representa a aquellos usuarios que están registrados en el sistema y, por lo tanto, se dispone de sus datos y su historia previa, pudiendo ofrecer unos contenidos personalizados en función de su perfil.

4. Administrador de tienda: Este actor se encarga de gestionar una tienda individual del *mall*.

5. Administrador de *mall*: Este actor es el principal responsable de la gestión del conjunto de tiendas que componen el *mall*.

2.2 Opciones

2.2.1 Opciones del actor Usuario

1. Visitar tienda: Un usuario que visita el *mall* tiene la posibilidad de navegar libremente por cualquiera de las tiendas que lo componen. Así, puede encontrarse en las páginas de una tienda y decidir cambiar a otra, pudiendo incluso realizar una navegación jerárquica a través de la estructura de tiendas.

2. Buscar productos: Un usuario pretende obtener una lista de productos que cumplan una serie de criterios establecidos en la búsqueda que se realiza. Las búsquedas que puede realizar un usuario pueden ser de

diferentes tipos: búsquedas a partir de una serie de palabras claves, búsquedas por una categoría determinada, ...

3. Consultar producto: El usuario quiere obtener información detallada acerca de un producto concreto. Se deben proporcionar al usuario todos los datos del producto, desde los datos propios del producto (nombre, descripción, precio, ...) hasta el conjunto de recursos asociados (imágenes, vídeos, ...), los productos relacionados, aquellos que han comprado otros usuarios junto con este, las críticas que se han realizado sobre el producto...

4. Buscar usuarios: Un usuario puede buscar a una persona determinada con la intención de regalarle algún producto de los que aparecen en su lista de deseos. Esta búsqueda la puede realizar introduciendo el nombre o apellidos del usuario buscado o la dirección de correo electrónico.

5. Consultar lista de deseos: El usuario puede visualizar aquellos productos de la lista de deseos de otro usuario que aún no le han sido regalados, y que por lo tanto le puede comprar.

6. Consultar noticia: El usuario puede visualizar información relativa a las últimas noticias que han aparecido en el servidor de noticias del *mall*.

2.2.2 Opciones del actor Usuario Anónimo

1. Darse de alta: El usuario introduce sus datos personales, quedando almacenados en el sistema de forma permanente.

2. Registrarse: El usuario introduce su identificador y su clave, y si la autentificación tiene éxito, será reconocido por el sistema.

2.2.3 Opciones del actor Usuario Registrado

1. Comprar productos: Un usuario elige una serie de productos que quiere comprar, se obtiene el importe total de los productos junto con los impuestos y gastos de envío, y se procesa el pedido para que se le pueda enviar al cliente cuando se considere oportuno.

2. Actualizar los datos personales: El usuario consulta sus datos personales, y modifica aquellos que sean erróneos, quedando los cambios almacenados en el sistema de forma permanente.

3. Cambiar de *password*: El usuario cambia el *password* con el que accede al sistema por otro nuevo.

4. Salir del sistema: El usuario termina de realizar sus operaciones y abandona el sistema.

5. Ver la lista de deseos: El usuario visualiza el contenido de su lista de deseos para ver el estado de los productos que quiere que le regalen.

6. Añadir un producto a la lista de deseos: El usuario selecciona un producto que le interesa y lo pone en su lista de deseos con la intención de que alguien se lo regale.

7. Ver los pedidos realizados: El usuario obtiene una lista con todos los pedidos que ha realizado en el *mall* junto con los datos de ese pedido y su estado actual (servido, no servido o cancelado).

8. Obtener información detallada de un pedido: El usuario visualiza la información relativa a un pedido concreto de forma detallada, es decir, cada una de las líneas que componen el pedido.

9. Cancelar un pedido: El usuario, si se arrepiente de haber realizado un determinado pedido y este aún no se ha servido, puede cancelarlo para que no se llegue a hacer efectivo.

10. Realizar una crítica de un producto: El usuario realiza un crítica de un producto que ha comprado o consultado y le da una valoración. Posteriormente, otro usuario podrá ver esta información, de forma que le ayude a decidir si compra el producto o no.

2.2.4 Opciones del actor Administrador de Tienda

1. Realizar el mantenimiento de los producto de la tienda: El administrador de tienda se ocupa de crear nuevos productos y de actualizar aquellos con datos erróneos u obsoletos.

2. Realizar el mantenimiento de las categorías de la tienda: El administrador de tienda se encarga de crear y actualizar las categorías en las que se clasifican los productos existentes en la tienda.

2.2.5 Opciones del actor Administrador de *Mall*

1. Visualizar pedidos pendientes: El administrador de *mall* obtiene una lista de todos aquellos pedidos que se han realizado y que todavía no han sido servidos.

2. Marcar un pedido como servido: El administrador de *mall* analiza los pedidos pendientes y si hay suficiente material en *stock* para cumplimentar un pedido, lo selecciona para envío y lo marca como servido.

3. Realizar el mantenimiento de los países del sistema: El administrador de *mall* se ocupa de crear nuevos países disponibles en el sistema y de actualizar aquellos con datos erróneos u obsoletos.

4. Realizar el mantenimiento de las formas de envío del sistema: El administrador de *mall* se ocupa de crear nuevas formas de envío dentro de cada país en el sistema y de actualizar aquellas con datos erróneos u obsoletos.

5. Realizar el mantenimiento de las formas de pago del sistema: El administrador de *mall* se ocupa de crear nuevas formas de pago en el sistema y de actualizar aquellas con datos erróneos u obsoletos.

6. Realizar el mantenimiento de los proveedores: El administrador de *mall* se ocupa de crear nuevos proveedores de material en el sistema y de actualizar aquellos con datos erróneos u obsoletos.

7. Realizar el mantenimiento de las noticia del servidor: El administrador de *mall* es el responsable de insertar y actualizar las noticias del servidor que serán accesibles para los usuarios a través de las páginas de la aplicación.

8. Consultar *stock*: El administrador del *mall* puede consultar en cada una de las tiendas aquellos productos que tienen su número de existencias por debajo de un determinado umbral.

9. Gestionar el *stock*: El administrador del *mall* se encarga de actualizar las unidades existentes de los productos de las tiendas cuando recibe nuevo material de los proveedores.

10. Consultar un proveedor de material: El administrador del *mall* puede acceder a los datos del proveedor de cualquier producto de sus tiendas con el fin de poder ponerse en contacto con él y pedir un reabastecimiento de material.

3 El Problema

El sistema de *comercio electrónico* desarrollado dispone de una gran variedad de opciones y facilidades, orientadas todas ellas a favorecer las compras de los usuarios. Incluso, si se desease, se podrían implementar nuevas opciones para aumentar las capacidades del sistema.

Pero estaríamos limitando mucho el sistema si nos centrásemos únicamente en las funcionalidades que la aplicación ofrece. Dado que se trata de un sistema abierto al gran público mediante Internet, habría que tener en cuenta los distintos tipos de usuarios que van a conectarse al sistema.

Este aspecto es especialmente relevante en el hecho de que, dados los continuos avances tecnológicos que se están produciendo en los últimos años, la gama de dispositivos electrónicos disponibles en el mercado es muy amplia y cada uno de ellos dispone de unas características propias que se adaptan a las condiciones de los diversos usuarios.

Al interactuar con el sistema de *comercio electrónico* se deberían tener en cuenta las características y capacidades de cada dispositivo de cara a sacarle un rendimiento óptimo a cada uno de ellos. Por ejemplo, un usuario que accede al sistema utilizando un PC con un navegador tradicional, puede visualizar una cantidad de información por pantalla muchísimo mayor que una persona que se quiere conectar mediante un dispositivo WAP. Además, también puede ser muy diferente la forma de navegación recomendable para cada uno de los dispositivos.

Pero dejando a un lado las diferencias físicas obvias entre algunos de los dispositivos que podrían utilizar los usuarios, hay que pensar también que el lenguaje que puede entender cada uno de estos dispositivos no tiene por qué ser el mismo, lo cual obliga a que la respuesta ante una petición sea distinta para cada uno de ellos.

De esta manera, el resultado que debería devolver el sistema ante una petición realizada por un dispositivo WAP (formato WML) es muy diferente de si la petición se recibe de un navegador tradicional (formato HTML). Incluso se debería poder construir una página HTML específica para cada navegador (Internet Explorer o Netscape), que a pesar de entender en teoría el mismo formato HTML, ofrecen ciertas diferencias que podemos desear controlar.

Este distinto comportamiento ante peticiones de distintos dispositivos, nos lleva a pensar que posiblemente habría que tener diversas aplicaciones corriendo, cada una de ellas atendiendo a peticiones de un dispositivo concreto. Sin embargo, dada la gran variedad de dispositivos existentes, esta solución no parece viable.

Además, la esencia del sistema de *comercio electrónico* es la misma para todos los casos, ya que el motor de la aplicación, y el contenido y funcionalidades que se pretenden ofrecer a los usuarios es común a todos ellos, por lo que parece un gasto excesivo e innecesario de recursos para resolver el problema que se nos plantea.

Por lo tanto, habría que intentar buscar una solución que nos permitiese utilizar un núcleo común del sistema, pero generando visualizaciones distintas que se adapten a las características particulares del dispositivo con el que se está realizando la petición.

4 Cómo se Resuelve

La solución al problema planteado consiste en definir cuatro subsistemas independientes que se encarguen de las siguientes funciones:

- El tratamiento de la petición y la obtención de la información y criterios necesarios para procesarla correctamente.

- La generación de un documento XML que contenga toda la información necesaria para proporcionar el resultado pero sin aportar ningún tipo de formato específico que esté orientado a una visualización concreta.
- Determinación del tipo de transformación a aplicar en función de las características de la petición recibida y del contexto.
- La transformación de este contenido general en la visualización deseada (transformación XSL) en función de los criterios determinados con anterioridad.

La arquitectura final del prototipo implementado puede verse en la figura 1. En este modelo se ofrece una descomposición del sistema global en una serie de subsistemas independientes que se intercambian mensajes.

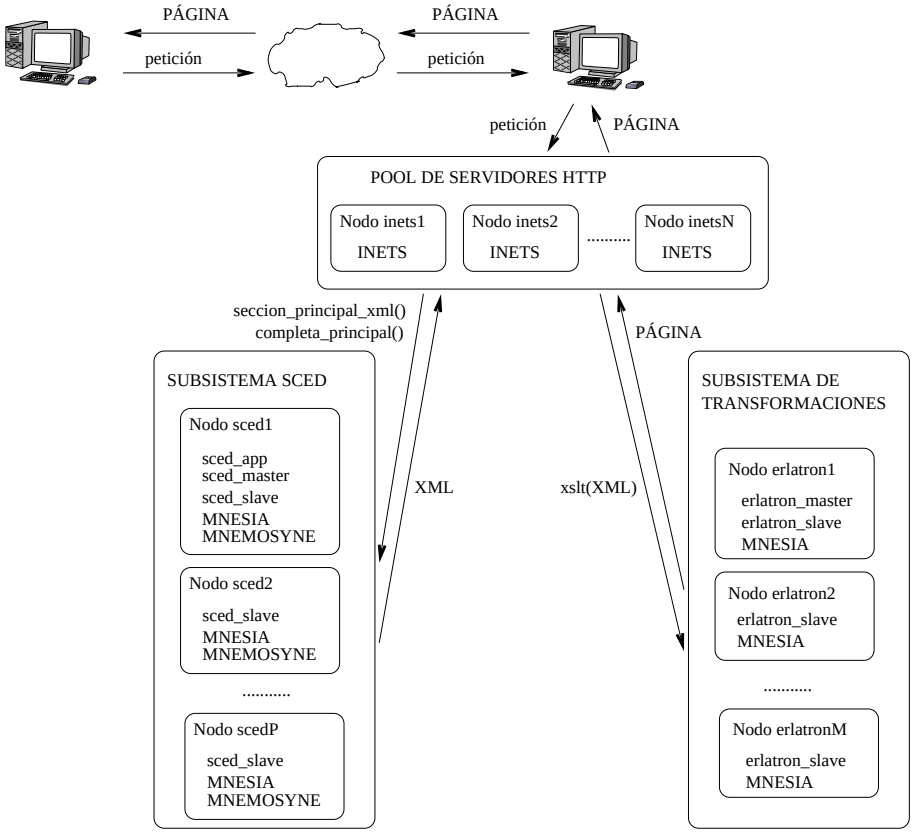


Figura 1: Arquitectura final del prototipo

En el sistema implementado existen tres subsistemas perfectamente diferenciables:

1. Subsistema de servidores Web.

   En un conjunto de nodos van a estar corriendo servidores HTTP (INETS). Este subsistema permitirá el acceso de los clientes al sistema de comercio electrónico.

   Las peticiones que llegan a cualquiera de los servidores HTTP, son interpretadas y redirigidas a los otros subsistemas. Como resultado se obtendrá la página solicitada en el formato deseado, que será enviada como respuesta. La existencia de varios INETS permite dotar al sistema de la redundancia necesaria para que, ante el fallo de cualquiera de los servidores, pueda seguir siendo accesible a través de cualquiera de los otros.

   Para conseguir que de cara al exterior parezca que el punto de entrada es único, se puede configurar una máquina para que funcione como *DNS Round Robin*, de manera que se encargue de redirigir las peticiones que le lleguen a los distintos servidores HTTP.

2. Subsistema `sced` (Sistema de Comercio Electrónico Distribuido).

   Sobre una serie de nodos va a estar corriendo la aplicación distribuida `sced_app`. Este subsistema es el encargado de construir el documento XML en función de los parámetros de la petición recibida y del contexto actual. Así, según la página en la que se encuentre un usuario y el perfil del mismo, este subsistema construirá documentos XML distintos.

   En cada uno de los nodos en los que corre esta aplicación, va a existir un `sced_slave`, que será el encargado de construir el documento XML. En uno sólo de esos nodos, va a estar el `sced_master` que será globalmente accesible, delegando las peticiones que reciba en los `sced_slave` del subsistema para que estos realicen el trabajo.

3. Subsistema de transformaciones (`erlatron`).

   Para la implantación de este subsistema se utilizó una aplicación disponible (`erlatron`), que va a estar corriendo sobre otra serie de nodos. Este subsistema se encarga de realizar la transformación del documento XML que construye el sistema `sced` aplicando una hoja de estilos XSL.

   El fichero XSL que debe aplicar lo elige en función de una serie de reglas, que dependen principalmente del navegador que esté usando el usuario y de la tienda en la que se encuentre.

   Al igual que en el sistema `sced`, existe un `erlatron_master` escuchando peticiones en uno de los nodos, y un `erlatron_slave` corriendo en cada uno de ellos, y con la responsabilidad de realizar la transformación.

El procedimiento para descargar una página es el siguiente:

1. Uno de los servidores HTTP recibe una petición.
2. Como resultado de la petición, se ejecuta un `erlet`.
3. Este `erlet` se encarga de coger de la petición toda aquella información relativa a la sesión y capturar la opción que se debe ejecutar.
4. Después se le pide al módulo correspondiente a la opción, que obtenga de la petición aquellos parámetros que necesita para llevar a cabo su funcionalidad, mediante la llamada a una función del módulo (`obtener_parametros`), de acuerdo con la interfaz que debe cumplir cualquier opción implementada.
5. Estos parámetros, junto con la información relativa a la sesión, se le pasan a otra función del módulo correspondiente a la opción que se a a utilizar (`seccion_principal_xml`).
6. Esta función, tiene la responsabilidad de elaborar el XML con el contenido necesario para construir la página que posteriormente visualizará el usuario. Además del XML, esta función devuelve el tipo de XSL que debe aplicarse para transformar el XML e información sobre la sesión (que será la misma salvo que la sesión haya terminado).
7. Este XML propio de la página se completa con el resto de contenido que se presentará al usuario, como pueden ser los menús o los demás complementos comunes a todas las páginas de la aplicación Web. Es decir, se completa el contenido del documento XML con información del contexto.
8. Con el documento XML y el tipo de XSL a aplicar, se llama al subsistema de transformaciones (`erlatron`) con una serie de parámetros del entorno (como puede ser el tipo de navegador que se esté utilizando), el cual se encargará de elegir el XSL más adecuado en el contexto actual y realizar la transformación.
9. No queda más que devolver el resultado de la transformación (que estará en el formato adecuado: HTML, WML, ...) a INETS, el cual se encargará de devolvérselo al navegador del cliente, ofreciéndole la visualización de la página solicitada.

5 Generalización en Forma de Patrón

La solución explicada para resolver el problema de la generación de múltiples visualizaciones se corresponde con el patrón Multi-Visualización, cuya estructura general se puede ver en la figura 2.

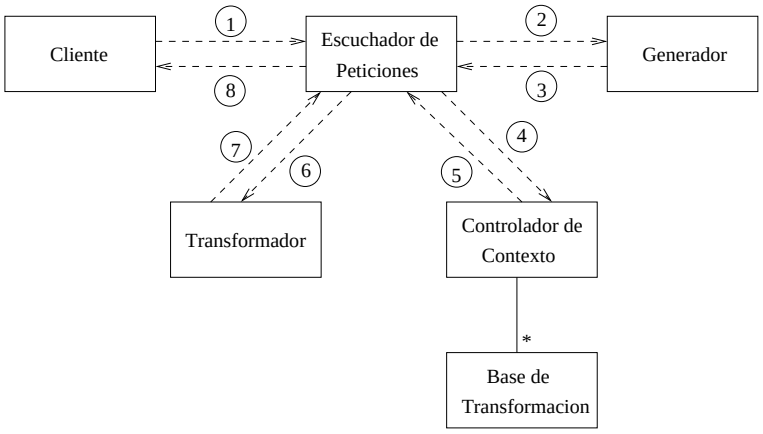


Figura 2: Patrón Multi-Visualización

En este diagrama, podemos distinguir los siguientes componentes:

- `Escuchador de Peticiones`: Se encarga de recoger las peticiones realizadas por el `Cliente`, ocupándose de procesarlas y obtener toda la información posible de la misma. Normalmente, con la petición vendrá información sobre la sesión que permitirá obtener datos necesarios para su procesamiento.

  Además, esta entidad es la encargada de coordinar el funcionamiento de los otros componentes, controlando los posibles errores de interacción entre los mismos (por ejemplo, la no disponibilidad de nodos encargados de construir el contenido o de aquellos que realizan la transformación).

- `Generador`: A partir de la información capturada por el `Escuchador de Peticiones`, este proceso se encarga de construir la respuesta en un formato estándar que contenga únicamente contenido, y nada de información relativa al aspecto que se le pueda dar a estos datos.

- `Controlador de Contexto`: A partir de la información capturada por el `Escuchador de Peticiones`, este proceso se encarga de determinar cuál es la `Base de Transformación` específica que debe aplicarse sobre el contenido construido con anterioridad.

- `Base de Transformación`: Entidad que permite, a partir del contenido generado en formato estándar, obtener una visualización en un formato específico.

- `Transformador`: A partir del contenido obtenido, este proceso aplica la `Base de Transformación` seleccionada por el `Controlador de Contexto` para obtener la salida en el formato deseado.

- `Cliente`: Entidad que interactúa con el sistema a través de peticiones esperando obtener el resultado en un formato acorde con sus características.

El funcionamiento de estos elementos es el siguiente:

- El `Cliente` interactúa únicamente con el `Escuchador de Peticiones`.
- El `Escuchador de Peticiones` se ocupará de procesar la petición e interactuar con el resto de los componentes para obtener el resultado en el formato adecuado, el cual será devuelto al `Cliente`.
- En primer lugar interactúa con el `Generador` pasádole la información recibida con la petición, pera que este construya el *contenido* de lo que será la futura visualización. Este *contenido* es equivalente para cualquier futura visualización y, por lo tanto, deberá estar construido en un formato estándar fácilmente transformable en otros formatos y que no contenga información sobre el aspecto que tendrá la salida (sólo tendrá datos sobre el contenido).
- A continuación, el `Escuchador de Peticiones` interactúa con el `Controlador de Contexto`, el cual es capaz de determinar a partir de la petición y la información del contexto cuál es la `Base de Transformación` que se debe aplicar. Normalmente, esto dependerá de la entidad que esté realizando la petición (`Cliente`), ya que la transformación tiene como finalidad dar un formato al *contenido* para que la visualización se adapte a sus características particulares.
- Una vez que tenemos el *contenido* que se quiere ofrecer y la *transformación* concreta que tenemos que aplicar, tan sólo tendremos que pasarle esta información al `Transformador` para que se encargue de aplicarla.
- El resultado obtenido tras realizar la transformación contará con el contenido que se desea ofrecer al `Cliente` pero estructurado en un formato apto para ser entendido correctamente por este. Por lo tanto, este resultado será el que se devuelva al `Cliente` y será específico para el tipo de dispositivo que haya hecho la petición.

Mediante la utilización de este patrón, se obtienen una serie de ventajas e inconvenientes como pueden ser:

- Ofrecer visualizaciones distintas a unos mismos contenidos, sin necesidad de duplicar elementos del sistema.
- División del proceso de generación del resultado de la petición en una serie de componentes independientes que pueden optimizarse de manera individual.

- Abstracción del tratamiento de peticiones en una serie de elementos con un único punto de entrada de cara al `Cliente`.
- Facilidad para la distribución en conjuntos de nodos de cada uno de los componentes explicados.
- Aumenta la complejidad del sistema.
- Ante la aparición de nuevos dispositivos, se pueden crear visualizaciones que se adpten a sus características sin necesidad de modificar el motor del sistema. Únicamente habrá que corregir el `Controlador de Contexto` para que sea capaz de detectar el nuevo dispositivo, y añadir una nueva `Base de Transformación`.

6 Adaptación del Patrón

Para una mayor comprensión de los elementos que costituyen el patrón, a continuación se van a identificar cada uno de los componentes genéricos del patrón, con aquellas entidades propias del caso particular estudiado.

En el sistema de *comercio electrónico*, utilizamos el patrón Multi-Visualización para estructurar todo el proceso de generación de las pantallas que visualiza el usuario. La estructura general del patrón adaptado a nuestro caso particular se puede ver en la figura 3.

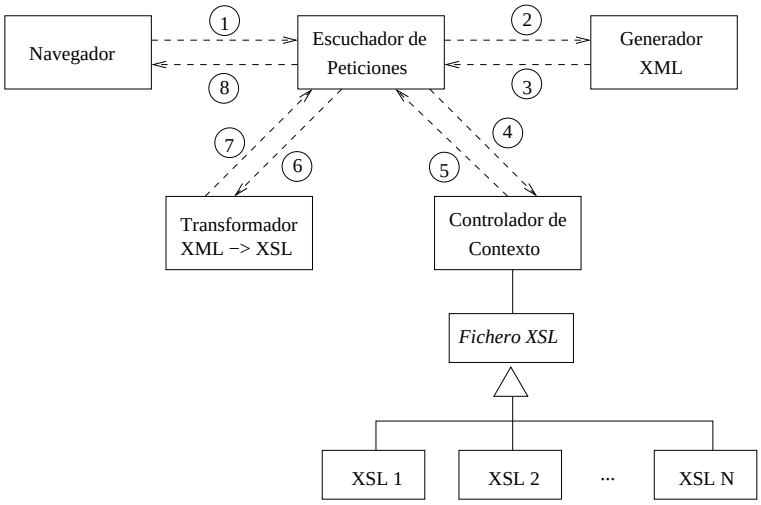


Figura 3: Generación de Páginas (Multi-Visualización)

En este caso, el `Navegador` (`Cliente`) realizará una petición sobre el servidor web (`Escuchador de Peticiones`) que mediante la invocación del

erlet coordinará todos los demás componentes de la arquitectura para generar el resultado.

El módulo de la opción hará las funciones de `Generador`, construyendo el documento XML con la información necesaria para la generación de la página.

Después, el propio *erlet*, obtendrá la información de contexto (`Controlador de Contexto`) a partir de la petición.

El `Controlador de Contexto` conoce a la familia de ficheros XSL (`Base de Transformación`) y los tiene clasificados según una serie de criterios determinados. Con la información obtenida de la petición que le pasa el `Escuchador de Peticiones`, es capaz de determinar el `Fichero XSL` adecuado.

La información de contexto y el documento XML se le envía al subsistema de transformaciones `erlatron` (`Transformador`) que procesará la petición y devolverá la página en el formato adecuado (HTML, WML, ...) para que pueda ser visualizado con el dispositivo usado para la conexión por parte del `Cliente`.

7 Conclusiones

Como puede verse, la utilización del patrón Multi-Visualización ha solucionado el problema de ofrecer distintas visualizaciones según el dispositivo que un usuario esté utilizando para conectarse al sistema de *comercio electrónico*.

Pero esta solución, aunque ha sido obtenida de un caso particular, no tiene una vinculación directa y única con las aplicaciones de *comercio electrónico*, sino todo lo contrario. Durante el proceso de análisis y explicación del patrón, se ha partido del caso particular del sistema estudiado para, posteriormente, realizar una generalizando hasta llegar a un nivel de abstracción, en el cual los distintos componentes y características del patrón extraído son totalmente independientes del caso particular de las aplicaciones de comercio electrónico.

Por lo tanto, se puede concluir que el patrón obtenido tiene unas características generales y, dado que ya ha sido probado que es útil para resolver un problema concreto, podrá ser utilizado para solucionar casos semejantes que surjan en el desarrollo de nuevos sistemas.

De esta manera, a partir de este momento, se dispone de una nueva herramienta que pueden usar los desarrolladores de *software* cuando se encuentren con el problema de implementar sistemas que ofrezcan diferentes visualizaciones para unos mismos contenidos.

Ante un problema similar, como ocurre con el resto de patrones, los desarrolladores de aplicaciones no tienen que plantearse nuevamente una solución, sino que como sabe que el patrón Multi-Visualización puede resol-

ver el problema, únicamente tendrá que adaptar sus componentes al caso particular del que se trate.

Por lo tanto, el uso del patrón Multi-Visualización, y en general de cualquier otro patrón, supone una reducción en el esfuerzo y en el coste del desarrollo de un sistema, y en una mayor claridad y reusabilidad del código generado, características muy importantes en cualquier ámbito del desarrollo de *software*.

Referencias

[AWWV95] J. L. Armstrong, M. C. Williams, C. Wikström, and S. R. Virding. *Concurrent Programming in Erlang*. Prentice Hall, 2nd edition edition, 1995.

[Dou03] Bruce Powel Douglass. *Real-Time Design Patterns: Robust Scalable Architecture for Real-Time Systems*. Addison-Wesley, 2nd edition edition, 2003.

[ERL01] Erlang otp documentation. www.erlang.org, 2001.

[Fer01] José Ramón Gulías Fernández. Desarrollo de un Sistema de Comercio Electrónico Ejecutable en un *Cluster* de Ordenadores. Proyecto Fin de Carrera. Dpto. de Computación. Facultad de Informática de A Coruña, Julio 2001.

[GHJV95] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns*. Addison Wesley, Reading, MA, 1995.

[Men98] Martin Menzow. *Comercio electrónico: Construcción de ciberalmacenes*. McGraw-Hill, first edition, 1998.

[PM99] Natanya Pitts-Moultis. *XML in record time*. Sybex, 2021 Challenger Driver, Suite 100, Alameda, CA 94501, USA, 1999.

[TS98] G. Winfield Treese and Lawrence C. Stewart. *Designing Systems for Internet Commerce*. Addison-Wesley, Reading, MA, USA, 1998.

[XML01] Xml bible. www.ibiblio.org/xml/books/bible/, 2001.

Sistema de gestión para un servidor de video bajo demanda*

Carlos Varela, Víctor M. Gulías, Alberto Valderruten, Carlos Abalde
LFCIA, Departamento de Computación, Universidade da Coruña
Campus de Elviña, 15071 A Coruña, España
email: {cvarela,gulias,valderruten,carlos}@dc.fi.udc.es

RESUMEN

En este artículo se describe un sistema de gestión para un servidor de vídeo bajo demanda usando SNMP. Permite por una parte la monitorización del servidor VoDKA, un sistema que proporciona servicios de vídeo a usuarios que pueden solicitar extractos de información en cualquier momento, y por otra parte soporta la gestión de cambios en el servidor, todo ello usando el estándar de gestión SNMP. Dicha gestión puede así realizarse a través de herramientas externas con independencia de la que se utilice. Para su implementación se explotan las posibilidades de Erlang/OTP, lenguaje funcional para desarrollar aplicaciones de tiempo real blando, tolerantes a fallos y distribuidas, usando el paradigma de paso de mensajes.

Palabras clave: Monitorización, Instrumentación, Programación Funcional, Erlang/OTP, Vídeo bajo Demanda, Evaluación del Rendimiento.

ABSTRACT

In this paper we describe a management system for a video-on-demand server based on SNMP. It allows the monitoring of the VoDKA server, a system which offer video services to users that can demand for data streams at any time, and also provides the management of the changes on the server, using the standard SNMP. Thus, the management can be done using any standard tool. The concurrent functional language Erlang/OTP has been chosen for the development because of its suitable features for the implementation of soft real-time fault-tolerant distributed processing applications, using message-passing.

Keywords: Monitoring, Instrumentation, Functional Programming, Erlang/OTP, Video on Demand, Performance Evaluation.

*Trabajo parcialmente financiado por CICyT, Proyecto TIC 2002-02859, *VRDADER: Verificación, Rendimiento y Disponibilidad de Aplicaciones Distribuidas en Entornos Reales* y por *Xunta de Galicia,* Proyecto PGIDT02TIC00101CT, *Diseño, construcción y validación de un sistema jerárquico de almacenamiento de alta capacidad, de bajo coste de adquisición y funcionamiento: CheapTB.*

1. Introducción

Un servidor de vídeo bajo demanda (servidor VoD, por sus iniciales en inglés) proporciona servicios de vídeo al usuario final, que puede solicitar la visualización de una secuencia de vídeo (películas bajo demanda, tele-enseñanza, compras desde el hogar, servicios de noticias interactivas, etc.) en cualquier momento, sin una previsión o planificación pre-establecida. La creciente demanda de este tipo de servicios condiciona el desarrollo de servidores VoD con un alto nivel de escalabilidad, tanto en capacidades de almacenamiento como en ancho de banda. Para ello ha propuesto en [1, 2] un sistema de almacenamiento jerárquico basado en un cluster de componentes de consumo con Linux. En su implementación se ha usado el lenguaje funcional concurrente Erlang [3], desarrollado por Ericsson, debido a las facilidades que ofrece para escribir aplicaciones de tiempo real complejas, tolerantes a fallos y distribuidas usando el paradigma de paso de mensajes.

Como base no sólo para la gestión del sistema distribuido durante su fase operacional, sino también para las etapas de validación y verificación del proceso de evaluación del rendimiento que se integra en la metodología de desarrollo, resulta fundamental disponer de los mecanismos de monitorización apropiados. Dichos mecanismos deben permitir la observación del sistema a diferentes niveles de detalle y se han de diseñar de forma que perturben en la menor medida posible el comportamiento funcional del sistema. Se ha recurrido a un conocido patrón de diseño, el *Observador* (*Observer* [4]), en el cual se desacopla el sujeto observado de aquellos agentes que observan sus evoluciones. Para afectar lo menos posible a la carga de cada uno de los sujetos, se introduce un agente mediador (*Mediator* [4]) local encargado de registrar los observadores, recibir las notificaciones de los sujetos que cohabitan en el nodo y distribuir los eventos a los observadores relevantes [5].

Pero para gestionar un servidor de vídeo esto no es suficiente. En este artículo se exploran las posibilidades de Erlang/OTP para el desarrollo de una infraestructura de gestión que pueda ser integrada en el servidor de vídeo bajo demanda VoDKA. El sistema construido permite la monitorización del estado de los recursos, la modificación del estado y la generación de datos estadísticos.

2. Gestión de redes

Las tecnologías de gestión de red se centran en la monitorización, interpretación y control de los comportamientos de la red. Los estándares de gestión de red buscan integrar estas funciones dentro de la heterogeneidad de dispositivos y protocolos que existen en los sistemas de red actuales.

Se pueden analizar los fundamentos de la gestión de red centrándose en las dos principales operaciones involucradas: monitorización y control. Aunque son conceptos de la gestión de redes, en este trabajo se van a aplicar a un servidor de *streaming* existente.

Monitorización

La monitorización se realiza observando y analizando el estado y el comportamiento de los sistemas. La información de *monitorización* puede clasificarse en *(i) estática*, cuando caracteriza la configuración actual y sus elementos, que varían con poca frecuencia, *(ii) dinámica*, cuando está relacionada con eventos que suceden en el sistema como un cambio de estado, y *(iii) estadística*, que incluye información que puede ser inferida a partir de la dinámica como la media de paquetes transmitidos por unidad de tiempo en un sistema.

En un sistema de monitorización se distinguen los elementos que se pueden ver en la figura 1. Los cuatro

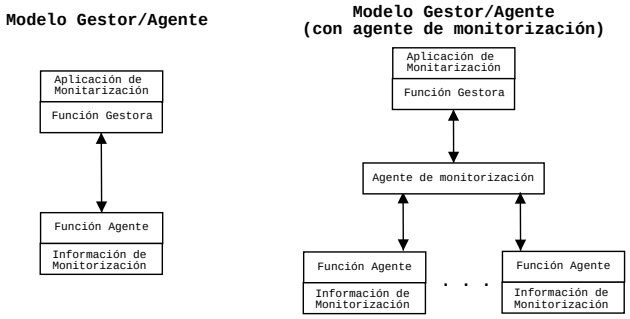


Figura 1: Esquema de un sistema de monitorización

componentes principales del sistema son: La *aplicación de monitorización*, que incluye las funciones de monitorización que son visibles para el usuario. La *función gestora*, que realiza la función básica de recuperar información de monitorización para la aplicación. La *función agente*, encargada de recolectar y almacenar información de uno o más elementos y comunicar la información al monitor. En cuanto a la *información de monitorización*, es la información que representa los recursos y sus actividades.

Opcionalmente, como se puede ver en la segunda parte de la figura, puede haber un módulo con información estadística, un *agente monitorizador*, que genera agregaciones y análisis estadísticos de la información. Si no está con el gestor, toma el papel de agente para comunicarse con él.

La información que es útil para la monitorización es recogida y almacenada por los agentes y se hace disponible para los sistemas de gestión, usando para esto último dos técnicas posibles: *polling* y *event-reporting*. El *polling* es una interacción basada en mensajes de petición/respuesta entre el gestor y los agentes. El gestor solicita parte o toda la información que posee un agente. El agente espera peticiones y devuelve información almacenada en su base de información de gestión. Un sistema de gestión puede usar *polling* para aprender acerca de la configuración de un sistema, para obtener información periódica o para investigar algo en detalle después de ser alertado de un problema. También es utilizado para generar respuestas a preguntas específicas de un usuario. Con el *event-reporting* la iniciativa, y con ello la complejidad, es del agente que, periódicamente o cuando se haya cumplido alguna condición, envía información al gestor. Aquí el gestor sólo tiene que configurar la actividad del agente, el periodo de recepción de informes o la condición para el envío de informes, y ponerse en espera.

Control

El *control* está vinculado con la modificación de los componentes de configuración que pueden provocar acciones predefinidas que son realizadas por esos componentes. El control se estructura en dos áreas: *control de la configuración* y *control de la seguridad*. El control de la seguridad se encarga de ofrecer seguridad en los ordenadores de una red para los recursos gestionados. El control de la configuración se centra en la definición de la información de configuración, el cambio de valores de atributos, la definición y modificación de relaciones entre los elementos del sistema, la inicialización y terminación de operaciones en el sistema.

3. SNMP

El término SNMP, *Simple Network Management Protocol*, se usa para referirse a una colección de especificaciones de gestión de red que incluyen al protocolo en sí, la definición de estructuras de datos y conceptos asociados [6]. El modelo de gestión de red usado para gestionar redes TCP/IP incluye los elementos *gestor*, *agente*, *base de datos de gestión (MIB)* y *protocolo de gestión*, relacionados entre sí según puede verse en la figura 2.

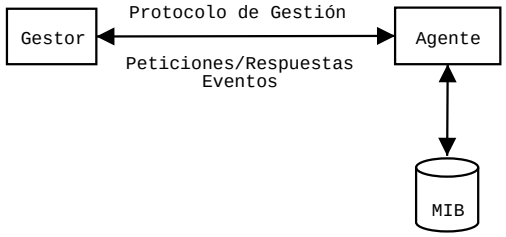


Figura 2: Conceptos básicos de los estándares de gestión

El *gestor* sirve como interfaz para la persona encargada de la gestión. Como mínimo debe incluir: (i) un conjunto de aplicaciones para el análisis de datos, recuperación de fallos ..., (ii) una interfaz para monitorizar y controlar la red, (iii) la capacidad de traducir los requerimientos actuales en la monitorización y control de los elementos remotos en la red y (iv) una base de datos con información de todas las entidades en la red. Sólo los dos últimos aspectos están sujetos a estandarización SNMP.

El *agente* es el otro elemento activo en el sistema de gestión de red. Suele ejecutarse en el dispositivo a gestionar o en una estación con acceso a los recursos gestionados. Responde a peticiones del gestor y puede asíncronamente enviarle información acerca de algún evento importante.

Los recursos de una red pueden ser gestionados representando esos recursos como objetos. La colección de objetos se llama *base de datos de gestión (MIB)*. Todos los objetos gestionados en el entorno SNMP están dispuestos en una estructura de árbol. Los objetos hoja son los objetos que se gestionan, donde cada uno representa algun recurso, actividad o información relativa a lo que tiene que ser gestionado. Además asociado con cada tipo de objeto en una MIB hay un identificador del tipo ASN.1 *Object Identifier* (en adelante, Oid) que es una secuencia de enteros. El identificador sirve para nombrar los objetos y para identificar la estructura de los tipos de objetos. Empezando desde la raiz del árbol, cada valor que compone el identicador de un objeto identifica una rama en el árbol. El gestor realiza la monitorización recuperando los valores de los objetos de la MIB. Asímismo puede realizar acciones o cambiar la configuración modificando valores de variables específicas. Están definidas en este momento dos MIBs estándar, conocidas como MIB-I y MIB-II [7].

Los agentes y el gestor se comunican por medio del *protocolo de gestión de red*. El protocolo utilizado para la gestión de TCP/IP es el *Protocolo Simple de Gestión de Red*, SNMP[8, 9], que incluye las capacidades siguientes: (i) *get*, para recuperar el valor de objetos en el agente; (ii) *set*, para establecer valores de objetos en el agente; y (iii) *trap*, para notificar al gestor eventos significativos.

En la revisión SNMPv3 se mejora la seguridad creando el modelo de seguridad basado en el usuario, USM[10], y el modelo de control de acceso basado en vistas, VACM[11].

4. Arquitectura de gestión

La arquitectura propuesta está basada en el modelo gestor/agente con agentes de monitorización. En cada nodo se instala un agente SNMP que recogerá la información estática y dinámica y atenderá las peticiones del gestor. Además, y para evitar que el gestor tenga que hacer peticiones a todos los nodos, se plantea construir una jerarquía de agentes de modo que cada agente contenga toda la información que posean los agentes que se encuentren por debajo de él. El agente también puede generar información estadística al margen de que otros sistemas puedan igualmente generarla. Esta arquitectura está representada en la figura 3.

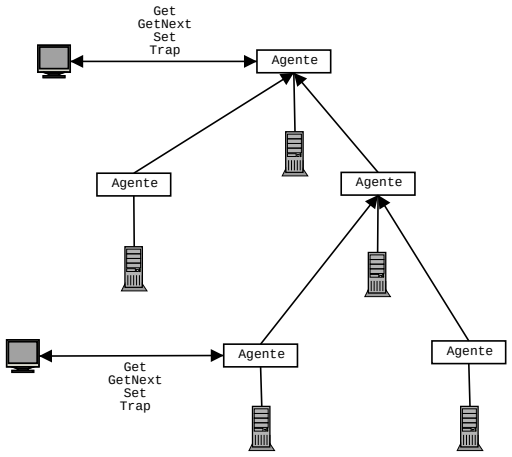


Figura 3: Arquitectura de gestión

En cada nodo el agente se comunica con el sistema VoDKA que allí se encuentre para conseguir la información que deba mantener en su MIB. El gestor es el encargado de pedir los datos a los agentes de forma periódica para actualizar el estado del sistema.

Un sistema VoDKA normalmente está desplegado en varios nodos por lo que será necesario más de un agente, uno en cada nodo. Los agentes mantienen la información en una MIB. Por lo tanto, los elementos que participan en el sistema de gestión son:

- El *agente*, que recoge información de VoDKA y contesta peticiones SNMP;
- La *MIB*, la base de información del agente, en la que guarda los datos que se necesitan para la gestión de VoDKA;
- El servidor *VoDKA*, el elemento que está siendo gestionado;

- El *Cliente SNMP*, encargado de consultar o modificar información en el agente.

Estos elementos se representan en el diagrama de clases de la figura 4. Queda por ver cómo es la estructura de la MIB y para eso hay que analizar la estructura de VoDKA para extraer su modelo de datos.

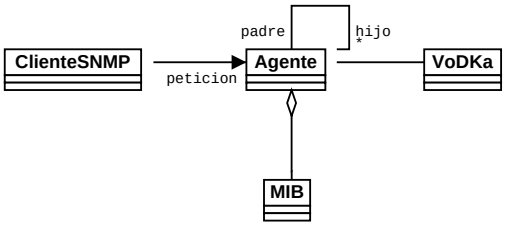


Figura 4: Diagrama de clases conceptual

5. Diseño del agente

El diseño del agente se realizó en varias etapas tras las cuales se fue incrementando la funcionalidad del mismo:

1. A partir del toolkit de ERLANG para la creación de agentes SNMP [12] y del mecanismo de paso de mensajes de ERLANG, se construyó un agente capaz de entender mensajes SNMP y de gestionar su estado interno de una forma sencilla.
2. Se diseñó un mecanismo que permitiese obtener la información que contiene el sistema VoDKA en un nodo y lo almanacenase en la MIB del agente.
3. Se diseñó la forma de crear la jerarquía de agentes.
4. Se introdujo el mecanismo que permite modificar el estado del servidor.
5. Por último se incluyó la posibilidad de dar de alta nuevas instancias en la MIB.

Primer agente

Para conseguir el primer objetivo lo primero que se necesita es un mecanismo cómodo que permita la inserción de valores en la MIB. Por cómodo se entiende que no sea necesario especificar los Oids de los objetos, ni conocerse el índice de las columnas de las tablas, como se especifica en los mensajes SNMP, sino que sea suficiente con dar su nombre. Algo del estilo:

```
insertar_fila(
   tabla_almacenamientos,
   [{pidTablaAlmacenamientos, Pid},
    {descripcionTablaAlmacenamientos, Nombre},
    {siguienteTablaAlmacenamientos, Siguiente},
    {estadoTablaAlmacenamientos, Estado},
    {tipoTablaAlmacenamientos, Tipo}]
);
```

para insertar una fila en una tabla, o

```
obtener_valores_fila_tabla(tabla_medios, Clave, [tamanoTablaMedios])
```

para obtener el valor de una columna en una tabla conociendo el valor de la clave.

Para ello se construye una capa *software* por encima de los módulos `snmp` y `snmp_index`, que forman parte del SNMP toolkit de ERLANG. Esta capa está formada por tres módulos que se pueden ver en la figura 5.

El módulo `tabla_mib` proporciona una implementación de una tabla para la MIB, y es la encargada de realizar las operaciones de modificación, inserción y validación de valores en la tabla. Este módulo es un módulo general y al crear una tabla con él hay que configurar ciertos parámetros que se pasan a la función que crea la tabla. Las funciones que exporta este módulo se pueden dividir en dos grupos: las que no usan Oids ni índices y las que sí los manejan.

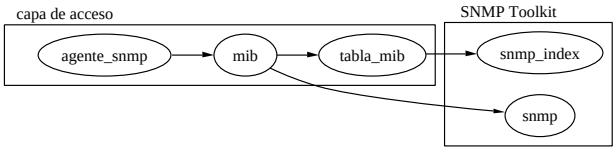


Figura 5: Relación entre la capa de acceso y el SNMP Toolkit

- **Funciones que no manejan Oids ni índices:**
  - `insertar_fila/2`: recibe como parámetros una lista de pares {`NombreColumna, Valor`} que representa una fila y una TablaMib. Inserta la fila en la tabla. Hay que tener cuidado de insertar **todas** las columnas (no importa el orden).
  - `eliminar_fila/2`: recibe como parámetros una tupla con los valores de la clave de una fila que será eliminada y una TablaMib.
  - `actualizar_fila/3`: recibe como parámetros una tupla con los valores de la clave, una lista como la que recibe la función `insertar_fila/2` y una TablaMib. Lo que hace es actualizar los valores de las columnas indicadas en la fila de la tabla que tenga esa clave.
  - `actualizar_fila_acumulando/3`: igual que `actualizar_fila/3` sólo que en lugar de sustituir los valores por los dados, éstos se suman a los ya existentes.
  - `obtener_fila_tabla/2`: recibe una clave y una TablaMib. Devuelve una lista con los valores de las columnas según el orden en que se definieron.
  - `obtener_valores_fila_tabla/3`: Igual que la anterior pero recibe un parámetro más para indicar los nombres de las columnas que se desean.
- **Funciones que manejan Oids y/o índices:**
  - `create/5`: recibe como parámetros un *atom()* que representa el nombre de la tabla, una clave del tipo *key_types()* de snmp_index, una lista con los índices de las columnas que forman el índice de la tabla, un *integer()* con el índice de la columna que es de tipo RowStatus y una lista de elementos de tipo *atom()* que representan los nombres de las columnas en el orden que el que están definidas en la MIB. Devuelve una TablaMib cuya representación interna no es necesario conocer.
  - `inserta_valores/3`: Recibe como parámetros un key() de snmp_index que representa el índice de la fila, una lista de tuplas {`Indice, Valor`} que representan los valores para cada columna y una TablaMib. Inserta las columnas en la tabla aplicando la función de inserción indicada al crearla. Devuelve {`noError, ValoresNoInsertados`} o {`genErr, Indice`} donde `ValoresNoInsertados` son los elementos de la lista de columnas que no se insertaron e `Indice` es el índice de la primera columna donde hubo un error.
  - `obtener_fila/2`: dado un Oid y una MibTable obtiene una lista con los valores de la fila correspondiente a ese Oid. Devuelve {`ok, {Fila, ColumnasIndice, RowStatusCol, NumeroColumnas}`} o `undefined` donde *Fila* es una lista con los valores de la fila, *ColumnasIndice* en una lista con los índices de las columnas que forman el índice, *RowtatusCol* es el índice de la columna con el RowStatus y *NumeroColumnas* es el número de columnas devueltas.
  - `obtener_siguiente_fila/2`: igual que la anterior pero obtiene la siguiente fila en orden lexicográfico.
  - `columna_row_status/1`: Obtiene el índice de la columna con el RowStatus.

El segundo grupo de funciones está pensado para ser usado por las funciones de instrumentación de la MIB.

Las funciones de instrumentación necesarias para el agente genérico proporcionado por el *toolkit* se definieron en el módulo `mib`. Este módulo, además, proporciona operaciones para manejar los objetos de las tablas sin necesidad de conocer su Oid. El módulo hace uso de tablas definidas con el módulo `tabla_mib` manteniendo una colección de ellas en su estructura interna. Para indicar la estructura de las tablas que se almacenan en la MIB se define un fichero que debe llamarse *tablas.txt* y está formado por términos ERLANG de la forma siguiente, utilizando variables que tienen el mismo significado y forma que las de la función `tabla_mib:create/9`:

```
{NombreTabla, Clave, ColumnasIndice, ColumnaRowStatus, NombresColumnas}
```

El módulo `mib` exporta las funciones siguientes, que buscan todas ellas una tabla en su estructura interna y delegan la operación correspondiente en ella:

- `insertar_fila/2`: recibe el nombre de una tabla y una lista de pares `{NombreColumna, Valor}` que representa una fila.
- `eliminar_fila/2`: recibe el nombre de una tabla y una tupla con los valores de la clave de una fila que será eliminada.
- `actualizar_fila/3`: recibe el nombre de una tabla, una tupla con los valores de la clave y una lista como la que recibe la función `insertar_fila/2`.
- `actualizar_fila_acumulando/3`: igual que `actualizar_fila/3`.
- `obtener_fila_tabla/2`: recibe el nombre de una tabla y su clave.
- `obtener_valores_fila_tabla/3`: igual que la anterior.

Además, el módulo exporta las funciones de instrumentación para el agente genérico del *toolkit* del ERLANG. Estas funciones son:

- `tabla(get, RowIndex, Cols, NombreTabla)`
- `tabla(get_next, RowIndex, Cols, NombreTabla)`

El módulo `agente_snmp` proporciona una fachada. A través de esta fachada se inicia el agente genérico de ERLANG y la MIB. Además permite un arranque remoto del agente. Este módulo exporta las funciones siguientes:

- `up/2`: recibe como primer parámetro un nodo y como segundo una lista de opciones, e inicia el agente en el nodo indicado.
- `insertar_fila/2`, `eliminar_fila/2`, `actualizar_fila/3`, `actualizar_fila_acumulando/3`, `obtener_fila_tabla/2`, `obtener_valores_fila_tabla/3`: delegan todas ellas en el módulo `mib`.

Consiguiendo información de VoDKA

Hay dos alternativas claras para obtener información del sistema VoDKA. Cada vez que le llega una consulta al agente, éste busca la información en el sistema por medio de la interfaz de los servidores genéricos (RGSs) que pueden hacer introspección para recuperar propiedades definidas dentro de ellos. O bien, se registra un observador en el sistema que escuche los eventos y almacene dicha información en el agente. La primera alternativa supone que cada vez que se realice una petición se explore todo el sistema (para sacar una "foto" del estado) y obtener la respuesta a partir de los datos recogidos. La segunda, en cambio, supone que se estén enviando datos al agente incluso en momentos en que esos datos no interesan porque no van a ser examinados.

Dado que el modelo de interacción con el agente es del tipo **petición/respuesta**, se puede suponer que habrá muchas consultas al agente por lo que la primera alternativa degradaría mucho el rendimiento del sistema, al ser una operación muy costosa consultar el estado de todos los RGSs, particularmente en un sistema distribuido. Se optó por registrar un observador y, para hacer esto, se desarrolló otra capa que se puede ver en la figura 6.

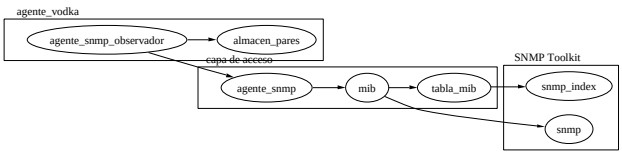


Figura 6: Módulos del agente

El módulo `agente_snmp_observador` proporciona una fachada que tiene el mismo comportamiento que `agente_snmp` y además se registra en el monitor de VoDKA para escuchar los eventos del sistema. Para obtener el mismo comportamiento que `agente_snmp` delega todas sus operaciones en ese módulo.

El módulo `almacen_pares` almacena pares de valores del tipo {`Valor1, Valor2`} y permite búsquedas sobre ellos. Su utilidad en el agente es la de almacenar los pares {*Pid,* {*Nodo, Nombre*}} en el caso de los *traders* y para los *pipes* que atienden una conexión {*PidPipe,* {*NodoConexion, PidConexion*}}. Este almacenamiento se necesita debido a que los eventos que envía el sistema al agente mandan el **pid** del proceso y las claves de los *traders* son el nodo en el que se encuentran y el nombre del *trader*. En el caso de los *pipes* interesa una forma rápida de conocer si un *pipe* atiende a una conexión para actualizar el progreso de la misma.

Para ver este comportamiento veremos dos ejemplos. En el primer ejemplo vemos qué sucede al darse de alta un almacenamiento en el sistema (figura 7).

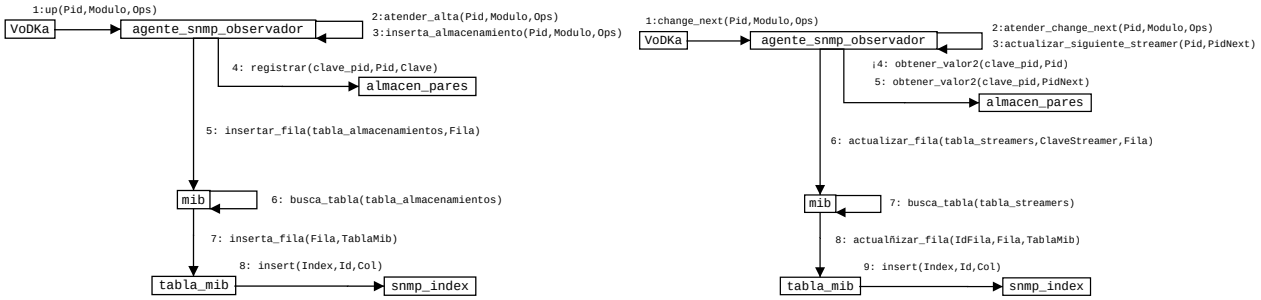


Figura 7: Alta de un almacenamiento y cambio del trader siguiente de un streamer

El sistema informa de un alta y envía al agente un evento de tipo **up** que contiene información como el pid, nombre del módulo y otras cosas que dependen del tipo de *trader* que se dé de alta. El agente recibe el evento y, tras analizarlo, concluye que es un almacenamiento y pasa a insertarlo en la MIB, pero antes de insertarlo registra el pid en un almacén de pares llamado *clave_pid* donde se guardará el pid junto con la clave del *trader* en la tabla de almacenamientos. Luego invoca la operación `insertar_fila` sobre la MIB indicando la fila y la tabla sobre la que quiere insertar, que en este caso es la tabla `tabla_almacenamientos`. Una vez localizada le indica a la tabla la fila que tiene que insertar, la cual realiza la operación a través del módulo `snmp_index`.

En un segundo ejemplo examinamos qué sucede cuando se modifica el *trader* siguiente de un *streamer* (figura 7). Aquí el sistema informa al agente de un cambio de *trader* siguiente en un *trader*. Después de analizar el evento llega a la conclusión de que el *trader* es un *streamer* y procede a actualizar la información que posee de éste, pero para lograr este fin necesita conseguir las claves de los dos *traders* y se las pide al almacén de pares. Un vez que conoce las claves le pide a la MIB que actualice en la tabla de *streamers* la fila que tiene la clave del *streamer*. La MIB busca la tabla y actualiza el contenido de la fila.

Jerarquía de agentes

Lo normal es que un sistema VoDKA se encuentre desplegado en varios nodos. Para minimizar el número de agentes a consultar para conocer el estado del sistema es conveniente poder definir jerarquías de agentes que permitan que la información de los agentes más bajos en la jerarquía se encuentre también en sus agentes inmediatamente superiores.

Para crear la jerarquía se mantienen referencias internas al padre que se pasan en el parámetro de opciones al crear el agente en la función `up/2`. En la lista de opciones hay que incluir la tupla {`parent, Pid`} donde *Pid* es el pid del agente padre. Una vez creado el agente éste informará al padre cada vez que altere el contenido de su MIB para que realice el mismo cambio en la suya. Podemos ver un ejemplo de esto en la figura 8. Aquí el agente que se encuentra en el nodo 1 conoce la información de todo el sistema por lo que un agente conectado a este nodo podría monitorizar todo el sistema. Un agente conectado al nodo 2 no sabría nada acerca del nodo 1 ni del nodo 3.

De este modo se consigue minimizar el número de consultas necesarias en un sistema grande; además como la notificación de los eventos es local en el nodo la información que se pasa a los padres (que normalmente se encontrarán en nodos distintos) ya es una información filtrada y en la mayor parte de los casos más pequeña que la información que envía el evento. Esto proporciona una alternativa mejor que registar varios observadores

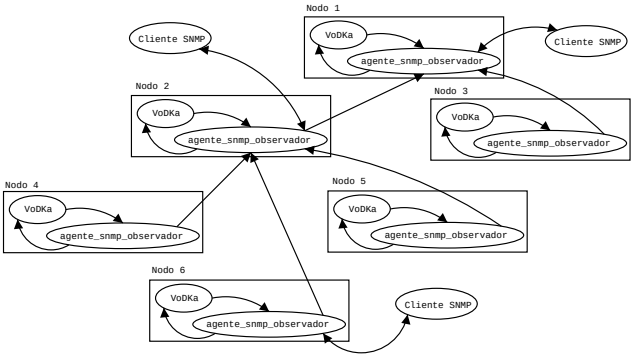


Figura 8: Configuración en varios nodos

en distintos nodos para un solo agente, al reducir el tráfico de gestión en la red.

Esta forma de mantener la información en los nodos introduce una restricción en las operaciones de inserción de nuevas instancias que viene dado por la forma en la que insertan datos usando SNMP. En primer lugar, con el sistema explicado hasta ahora no hay nada que impida que desde un nodo se dé de alta un *trader* en otro nodo. Por otro lado una de las formas comunes de dar de alta una fila de una tabla en la MIB de un agente es invocar operaciones `get` o `getnext` hasta encontrar una clave no utilizada. Si se hace esto en el nodo A y se da de alta una fila en otro nodo B puede ser que el nodo A no se encuentre por debajo en la jerarquía y sin embargo la clave de la fila que se intenta dar de alta ya exista, en cuyo caso se debería enviar un error informando de que no se puede usar.

Modificar el estado del servidor

La modificación de valores sucede en dos fases. Primero se validan los valores para ver si son correctos: por ejemplo, si se modifica el grupo de cachés al que pertenece una caché deberá mirarse si ese grupo caché existe. Después se hace la modificación en el sistema.

Las modificaciones no se hacen directamente sobre la MIB; en lugar de eso se modifica el sistema realizando las operaciones necesarias y el mecanismo de eventos visto antes informará al agente del cambio y en ese momento se reflejará dentro del agente.

Para permitir modificaciones hace falta crear 3 nuevas funciones de instrumentación:

- `tabla(is_set_ok, RowIndex, Cols, NombreTabla)`: comprueba que los parámetros están bien, en caso de no estarlo se llama a **undo**, sinó a **set**. Aquí también se pueden reservar recursos para hacer la modificación, ...
- `tabla(set, RowIndex, Cols, NombreTabla)`: realiza la modificación de los parámetros.
- `tabla(undo, RowIndex, Cols, NombreTabla)`: libera recursos solicitados por **is_set_ok** y deshace los cambios que haya realizado.

El problema que nos encontramos es que en cada tabla de la MIB hay que realizar unas validaciones y unas acciones distintas. La solución está basada en el patrón *Estrategia* (*Strategy* [4]). Lo que nos gustaría es que cada vez que se invoque **is_set_ok** o **set** se hicieran las comprobaciones o las modificaciones que corresponden a la tabla indicada por *NombreTabla*. Para esto primero se definen dos nuevas funciones en el módulo `tabla_mib`:

- `modifica_valores/3`: Recibe como parámetros el Oid de la fila, una lista de tuplas {`Indice, Valor`} y una TablaMib. Modifica la fila indicada por el Oid cambiando los valores indicados en la lista de tuplas, en la tabla *MibTable*. Devuelve {`noError,0`} en caso de no haber errores o {`genErr, Indice`} donde `Indice` indica la columna que provocó el fallo.
- `valida_valores/3`: recibe los mismos parámetros que la anterior pero esta vez valida las columnas. Devuelve {`noError,0`} o {`wrongValue, Indice`} donde `Indice` indica la primera columna que no pasó la validación.

El mecanismo que permite que cada tabla valide las columnas que quiera y de la forma que quiera es el siguiente: Se definen unas funciones de validación que deben cumplir los siguientes requisitos:

- Tener al menos un parámetro que será el Oid de la fila. Este parámetro será siempre el primero.
- Devolver `noError` en caso de que sea correcto y {`wrongValue, NumParam`} en caso de que un parámetro no pase la validación siendo el primer parámetro (el Oid) el 0.

La función `valida_valores/3` llamará a estas funciones para validar las columnas en cada validación de los valores de una fila. En caso de que una sola validación dé error devolverá {`wrongValue, ColumnaFallo`}.

Para modificar los parámetros se definen funciones de modificación que deben cumplir:

- Tener al menos un parámetro que será el Oid de la fila. Este parámetro será siempre el primero.
- Devolver `noError` en caso de que sea correcto y `genErr` en caso de haber un error cualquiera.

Las funciones de modificación serán llamadas para modificar las columnas de una fila por la función `modifica_valores/3`. En caso de que una sola modificación dé error devolverá {`genErr, ColumnaFallo`}.

Sólo queda asignar estas funciones a cada tabla y dentro de cada tabla saber para qué columnas se aplican. La forma de hacerlo es indicar en la creación de la tabla las dos listas de funciones indicando antes de cada función los índices de las columnas que se pasarán como parámetros. Para eso se añaden dos parámetros a la función `create/5` dando lugar a la función `create/7`, estos parámetros son: una lista de funciones de modificación formada por tuplas con la forma {`Indices, Modulo, Funcion`} donde se indican los índices de las columnas sobre las que se aplica la función, el módulo en el que se encuentra la función y el nombre de la función; donde las funciones deben devolver `noError` en caso de no haber error o `genErr` en caso contrario, y otra lista de funciones de validación, igual que la anterior salvo porque las funciones deben devolver `noError` en caso de no haber error o `genErr` en caso contrario.
Estas nuevas funciones se introdujeron en un módulo llamado `estrategias` quedando los módulos como se ilustra en la figura 9.

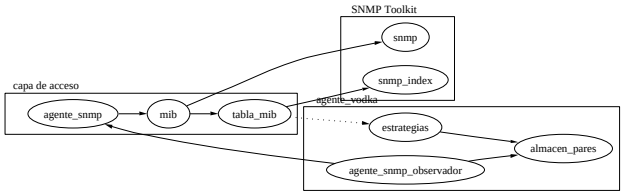


Figura 9: Módulos del agente

Vamos a ver un ejemplo para mostrar el funcionamiento de este mecanismo (figura 10). El cliente quiere

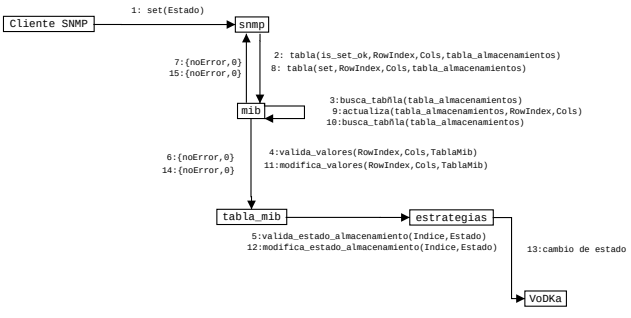


Figura 10: Modificación de un valor

modificar el estado de un almacenamiento e invoca una operación `set` desde el cliente SNMP sobre la tabla de almacenamientos indicando el Oid de la fila. El agente SNMP genérico llama a nuestra función `is_set_ok`, en la MIB se busca la tabla y se llama a `valida_valores/3` que invoca la operación de validación asignada a esa columna, que en este caso es `valida_estado_almacenamiento/2`. El resultado es positivo y se envía un mensaje

con {noError,0} a la MIB que a su vez se lo envía al agente SNMP genérico. Al no haber error se invoca a nuestra operación set que provoca una llamada a modifica_valores/3. modifica_estado_almacenamiento/2 es la función que modifica el estado, esta función provoca una llamada al sistema VoDKA para alterar el valor. Si la llamada es correcta se informa y finalmente el agente genérico muestra al cliente los valores insertados. No se modifica realmente el contenido de la MIB, lo que se hace es provocar un cambio en el sistema.

Alta de nuevas instancias

Finalmente queremos poder dar de alta nuevas instancias en la MIB. Se sabe que se da de alta una nueva instancia porque la columna con el *RowStatus* tiene el valor active (6). La solución es análoga a la anterior solo que en este caso se define una única función de inserción y se introduce la posibilidad de indicar los valores por defecto que se usan en caso de no dar valores para algunas columnas obligadas.

Las funciones de inserción deben tener al menos un parámetro que será el Oid de la fila. Este parámetro será siempre el primero. Y devolver noError en caso de que sea correcto y genErr en caso de haber un error cualquiera. Estas funciones serán llamadas para insertar las columnas de una fila por la función inserta_valores/3. En caso de que la inserción dé error devolverá {genErr, 0}, si no hay error devuelve {noError, ValoresNoInsertados} donde ValoresNoInsertados son los elementos de Cols que no fueron insertados, en este caso la MIB tiene la responsabilidad de llamar a {modifica_valores/3} con esta nueva lista para completar la operación.

Se añaden dos parámetros a la función create/7 dando lugar a la función create/9, estos parámetros son: una lista de funciones de inserción con la misma forma que las otras listas de funciones, si se hace una operación **set** con más columnas que las necesarias para la inserción (i.e. al dar de alta un *trader* no se puede especificar el siguiente trader) después de la inserción se invocan las operaciónes de modificación sobre los parámetros restantes; las funciones deben devolver noError en caso de no haber error o genErr en caso contrario, y una lista de valores por defecto formada por pares {Indice,Valor} que indican el valor por defecto en una inserción para la columna con índice Indice en caso de que no se dé ningún valor para ella.
Veremos qué sucede al dar de alta una caché desde un cliente SNMP (figura 11).

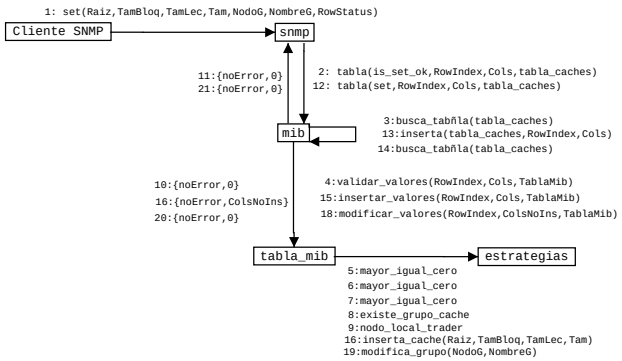


Figura 11: Nueva instancia en la MIB

El cliente quiere dar de alta una caché y especifica la raíz de la caché, el tamaño de bloque, el tamaño de lectura, el tamaño de la caché y el grupo al que pertenece. Como ya vimos, este tipo de operaciones se realiza en dos fases: una de validación de parámetros y otra en la que se hace la modificación o inserción en caso de que la validación sea correcta. El módulo snmp inicia la primera fase invocando la operación tabla(is_set_ok, RowIndex, Cols, tabla_caches) sobre la MIB. La MIB busca la tabla y le pide que valide los parámetros que envió el usuario. La tabla valida los parámetros uno a uno con las funciones de validación, en este caso se comprueba que los tamaños sean mayores que cero, que exista el grupo caché y que se trate de dar de alta en el nodo en el que se encuentra el agente. La operación es exitosa y se informa de ello a la MIB que avisa al agente. Entonces se inicia la segunda fase invocando la operación tabla(set, RowIndex, Cols, tabla_caches). Al estar presente la columna con el *RowStatus* en la operacion **set** y tener el valor active sabe que es una operación de inserción, busca la tabla y le pide que inserte la fila con la clave especificada por el usuario. La tabla invoca la operación inserta_cache que no usa todos los parámetros porque al dar de alta una caché no se puede especificar el grupo al que pertenece, devuelve a la MIB la lista de las columnas que no pudo insertar y ésta

invoca la operación de modificación sobre estas columnas. Finalmente la tabla modifica el grupo caché e informa de que todo fué bien. El agente informa al cliente de las columnas insertadas.

6. Conclusiones

En este trabajo se ha propuesto un diseño para monitorizar y gestionar los cambios del servidor de vídeo con una tecnología basada en la gestión de redes. El uso de un protocolo estándar como SNMP permite centrarse sólo en la adquisición de datos (y generación en caso de datos estadísticos) y en la implementación de las acciones posibles que se permiten realizar, dejando a las aplicaciones de gestión el resto.

El sistema desarrollado es lo suficientemente versátil como para permitir el control del servidor de forma remota y si se utilizan las posiblidades de seguridad de SNMPv3 se puede llegar a realizar una administración completa y segura del servidor usando SNMP.

El uso de patrones de diseño y de ERLANG/OTP para la implementación proporciona una opción robusta para conseguir un sistema distribuido y escalable, ejecutable sobre un *cluster* de ordenadores. Además el uso de este lenguaje permite dotar, de forma sencilla, de tolerancia a fallos a la aplicación desarrollada. El uso de las herramientas del lenguaje reducen de forma drástica los tiempos de construcción de un agente SNMP.

En la actualidad se plantea el uso de este sistema para gestionar los cambios en la arquitectura del servidor en base a resultados previstos por modelos de desempeño que se integren en el proceso de toma de decisiones.

Referencias

[1] M. Barreiro, V. Gulías, J. Sánchez, and J. Jorge, "The tertiary level in a functional cluster-based hierarchical vod system.," in *European Computer Aided Sustems Theory EUROCAST'01*, ISBN 84-699-3971-8, (Las Palmas de Gran Canaria), pp. 174–176, February 2001.

[2] J. Sánchez, V. Gulías, A. Valderruten, and J. Mosquera, "State of the art and design of vod systems," in *International Conference on Information Systems Analisis*, SCI'00-ISAS'00.ISBN 980-07-6694-4, (Orlando, USA), pp. 174–176, July 2000.

[3] J. L. Armstrong, M. C. Williams, C. Wikström, and S. R. Virding, *Concurrent Programming in Erlang*. Prentice Hall, 2nd edition ed., 1996.

[4] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns*. Addison Wesley, 1995.

[5] A. Valderruten, V. Gulías, J. Sánchez, and J. M. J.L. Freire, "Implementación de un modelo de monitorización para un servidor de vídeo bajo demanda en erlang," in *CLEI*, (Mérida, Venezuela), 2001.

[6] W. Stallings, *SNMP, SNMPv2, SNMPv3 and RMON 1 and RMON 2 Third Edition*. Addison-Wesley, December 1998.

[7] K. McCloghrie and M. Rose, "Management Information Base for Network Management of TCP/IP-based Internets: MIB-II; RFC-1213," *Internet Request for Comments*, no. 1213, 1991.

[8] J. Case, M. Fedor, M. Schoffstall, and J. Davin, "A simple network management protocol (SNMP); RFC-1157," *Internet Request for Comments*, May 1990.

[9] J. Case, K. McCloghrie, M. Rose, and W. S., "Protocol Operations for version 2 of the Simple Network Management Protocol (SNMPv2); RFC-1148," *Internet Request for Comments*, April 1993.

[10] U. Blumenthal and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3); RFC-2274," *Internet Request for Comments*, January 1998.

[11] B. Wijnen, R. Presuhn, and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP); RFC-2275," *Internet Request for Comments*, January 1998.

[12] "Erlang OTP Documentation/Simple Network Managent Protocol (SNMP)." http://www.erlang.org, 2002.

[13] M. T. Rose and K. McCloghrie, "RFC 1155: Structure and identification of management information for TCP/IP-based internets," May 1990.

[14] P. Y. Yemini, "A Critical Survey of Network Management Protocol Standars," in *Telecomunications Network Management into the 21st Century* (I. Press, ed.), ch. 2, 1994.

Analysing Participant's Interactions in Collaborative Learning Environments

Sandra de A. Siebra[1,2], Ana Carolina Salgado[1], Patrícia A. Tedesco[1]
[1]Universidade Federal de Pernambuco (UFPE) - Centro de Informática
Recife – PE, Brasil, Caixa Postal 7851 – Cidade Universitária - 50732-970
[2]Faculdade Integrada do Recife (FIR)
Recife – PE – Brasil
E-mail: {sas, acs, pcart}@cin.ufpe.br

Abstract

Collaborative learning can be motivated via environments that provide tools for communication, and discussion. In such environments, both students and instructors need online support in order to produce useful interactions. In this paper we discuss a novel method to analyse participant's interactions in Collaborative Learning Environments. We also propose an argumentation model to organise the group interaction and store the information in a multidimensional structure, which will be explored using analytical queries. An agent society receives information from the Collaborative Environment, reasons over it, and sends the results to the multidimensional structure. The results of this process generate the Learning Interaction Memory, which can be used by teachers to assess the learning progress and learners to evaluate their own progress.

Keywords: Collaborative Systems, CSCL, Context, Argumentation Models, Collaborative Learning

1. Introduction

There are several proposals of Internet-based learning tools (e.g. [16, 2]) that provide a wide variety of resources to learners, such as: information exchanging, non-linear navigation on learning material, discussion rooms and other communication tools (e.g., forums, blackboards, chats and video conferences). Despite the fact that these resources are important to learning processes, most current systems do not provide any intelligent support for students. For instance, these learning environments do not suggest better use of their tools so that learners can improve their knowledge and reflect about it [19, 4]. Such support is often given by teachers or human tutors during pre-defined times, and usually in response to explicit student requests. However, it is also the case that teachers generally do not have enough information to decide when is the best moment to interact with students, or to evaluate their learning process [23].

One of the ways of improving teacher-student relationship in virtual classrooms is to provide to the teacher information about the students' learning styles, progress and common doubts. Such information can be used to support learner reflection and can be acquired through the evaluation and diagnosis of interactions that occur among students and/or students and teacher. Hence, analysing the interactions will allow actions such as: (1) finding out both the most difficult and most interesting subjects for a specific student or group of students; (2) following the learning curve of students, finding the quicker learners in each subject, so that they can be used as peer tutors for students in trouble; (3) detecting apathetic students who need to be motivated.

However, evaluation, organisation, storage and management of information related to the interactions that occur in collaborative learning environments are complex tasks. Information can change at each new interaction with the collaborative environment. Furthermore, information is also context dependent (for example, access time, user's knowledge level and the role that an user was playing at the moment of the interaction are important for understand the ongoing discussion). In point of fact, learning always takes place in dynamic environments, characterised by a collection of relevant conditions and surrounding influences that make a situation unique and comprehensive [16].

This multidimensional aspect of interactions has motivated the use of data warehouses [22] in our project to store the content of such interactions. Storing this information into a multidimensional structure enables the system to analyse the interaction through On-Line Analytical Processing (OLAP) [20], where the interactions can be explored in different dimensions (e.g. subject, student knowledge level, etc) and levels of detail.

In order to be able to store the interactions for further analysis, we need to structure them, with the help of a data model. This generates what we call the Learning Interaction Memory (LIM). To model the LIM, we propose an extension of the IBIS (Issue Based Information System) argumentation model [11] to support learning interactions. And all these elements (argumentation model, set of results of the analytical queries etc) will be manipulated by an agent-based architecture.

Due to our use of argumentation models to organise information changed in learning environments, our work can be inserted in the Computer Supported Collaborative Argumentation (CSCA) area [13]. CSCA is an increasingly popular area of research, whereby networked computer-based tools are used to support the argumentation process. The key technologies attempt to achieve knowledge transformation (as opposed to simple representation or retrieval) by structuring discussions to graphically indicate agreements, disagreements etc. CSCA embeds the advantage to eliminate social context cues and provides a direct channel to express one's opinions.

The remainder of this document is structured as follows: section 2 describes some issues that are relevant to our work. Section 3 presents our tool for interaction analysis in collaborative learning environments, its agent-based architecture and the argumentation model proposed. Section 4 gives a brief overview of related works. Section 5 presents our conclusions and ideas for further work.

2. Relevant Issues

Computer Supported Collaborative Learning (CSCL) is a research area that investigates how the technology can support learning processes underlying the joint efforts of students working on specific tasks [27]. Collaborative learning encourages students who have a mutual aim of acquiring knowledge (and helping their peers to acquire) to interact with instructors and other students.

Computer Supported Collaborative Learning environments (CSCLEs) have the interaction as the key element to understand the process of knowledge building and the role of each student in that process. The analysis of the interaction can provide ways to support the individual and/or group needs, as well as improve the evaluation of different aspects (such as attitudes, weaknesses, knowledge level) of group/individuals behaviour. In order to analyse the interactions occurred into CSCL environments, we investigated argumentation models [11] as a way to structure these interactions and a data warehouse to store structured interactions, considering their multidimensional aspect. This section discusses these issues in detail.

2.1 Argumentation Models

Proper communication and orientation are necessary (but not sufficient) requirements for efficient collaborative learning,. Empirical studies [8] show that additional structured guidance (e.g. the use of an argumentation model can help students focus on the task) is often beneficial for learning.
Argumentation models have often been applied as a mean of systematising the communication patterns between members of a group [3]. The use of an argumentation model (for instance, IBIS [11]) allows the classification of all elements of discussion in pre-established abstractions of the model (e.g. Argument, Statement, etc) and their connection via a set of relations also pre-defined (e.g. provokes, generates, etc.). This is the case, for example, of Belvedere [10], Quorum [14], ARCoPAS [9] and SISCO [5].
In our work we use an argumentation model to categorise the message exchanging among participants and also to organise the information that composes the interactions in a collaborative environment, so that it can be used to:

- Assist students´ learning,
- Support instructors in: (a) evaluating students´ learning strategies; (b) assessing students during the whole learning process; (c) elaborating new courses or reformulating existing ones; and (d) analysing the roles students are playing (e.g., tutor, non-motivated student, shy student, etc.) so that effective help can be provided, taking into account the needs of the group but keeping in mind the problems of each of its members.
- Promote reflection in both students and instructors.
- Provide a well-organised group memory, similar to a smart FAQ. This feature allows participants of specific contexts to take advantage of the past-experiences of other group members.
- Store the identified information into a suitable structure, taking into account that the interaction within a collaborative environment is multidimensional. In other words, issues such as access time, access locals, user knowledge level, role that users are playing during the interaction, etc. are relevant to quantify (or qualify) the information which composes the LIM and will be used in the analysis process.

2.2 Multidimensional Modelling

Issues such as access time, access location, user's knowledge level and user roles are relevant to qualify the information in order to further analyse the interactions occurred in a collaborative environment. Thus, each interaction has a multidimensional facet that needs to be considered during its organisation.
According to several authors [15, 22, 17], multidimensional modelling is the most frequently used technique in data warehouse design. This has motivated us to consider this kind of data structure to store the Learning Interaction Memory (LIM). Other features that make the use of data warehouse suitable for our project are: subject-oriented approach (depending on the dimension considered at each moment), integrated, non-volatile and variable during the time [20]. The time variation needs to be considered because a student that presented a specific behaviour in situation X, at a time Y, can present a different behaviour in that same situation X at a time W.
Using operations like accessing, detailing and crossover, the information stored in the data warehouse can be used to support the instructor by evaluating and helping his students, working like a workgroup memory. Likewise, the system can use such information to support the learner reflection. Such operations will be implemented by OLAP (On-Line Analytical Processing) [22, 8], allowing the system to explore the interactions in different levels and dimensions (such as time, context and student level). Using OLAP operators we can do qualitative ("Which are the most frequent problems that have been found during the learning process?") and quantitative ("How many explanations have one question provoked?") analysis.

3. An Analytical Environment for Collaborative Learning

The socio-constructivist learning approach [25] suggests that the learner is part of a social group and s/he needs to be able to question, discover and understand the world around her/him. Current communication technology has made the implementation of computational environments that support and encourage group interactions possible at a reasonable cost.
In such situations, teachers can analyse the interactions, encouraging the participation of students and stimulating cooperative problem solving. The complexity of such a task is proportional to both the number of participants and the quality of their interaction. To provide information to instructors (so that they are able to follow the evolution of students in the short, medium and long terms), it is necessary to analyse users' actions in the learning environment,

as well as to evaluate which relevant pieces of knowledge have been shared and how they can help group members (in an individual or collective way) to perform better.
In next sections we present our agent-based system architecture that organises, stores and analyses information related to the interactions that occur in collaborative learning environments and the argumentation model proposed to structure the LIM.

3.1 System's Architecture

In order to be able to analyse group interactions, our tool will be connected to a CSCL environment. More specifically, our tool will communicate with the User Interface, Student Evaluation Modules and Student Model, all standard components (always present) of CSCLEs.
Figure 1 shows our tool's agent-based architecture [31]. It consists of an Agent Society composed by three agents: Perceptor, Modeller and Action Agents. Two models are also present: the Argumentation and the User Models. The Agent Society stores structured informations in a Learning Interaction Memory (LIM) implemented using a data warehouse to allow Analytical Queries and to keep historical informations about the discussions in the CSCL environment.

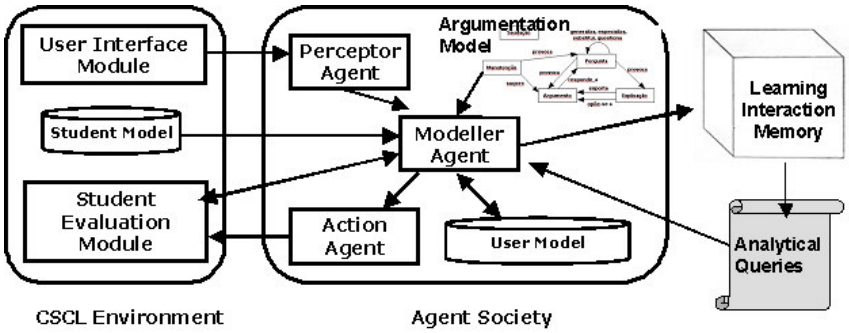


Fig. 1. Interactions Analysis System Architecture

The Perceptor Agent is responsible for monitoring all user interactions occurred in the learning environment and forwarding them to the Modeller Agent. The Modeller Agent is responsible for supporting students´ reflection as well as teachers´ decisions. It has three main functions: (1) uploading the data warehouse, based on the Argumentation Model adopted for the users' environment; (2) discovering the best strategy to adopt in order to promote reflection, based on the interactions analysis (through Analytical Queries to the LIM where user interactions are stored) and users´ models queries (to the CSCL environment's Student Model and Agent Society's User Model); (3) managing the Agent Society as a whole. Finally, the Action Agent is responsible for sending informations to the CSCL environment.
The Agent Society's User Model will be model through the following stereotypes: challenger, agreer, remiss, tutor, contributor, questioner, and unnatentive. Students will be classified into one of these categories according to their participation in the interactions environment.
The Modeller Agent needs to access structured interactions to execute its activities. In order to structure the interactions we could have used other dialog models, like dialogue games, as used in [30], or sentence openers, as used in [29]. However, as opposed to argumentation models, such models do not define dialogue episodes as precisely as we need in data warehouses. When a student use an abstraction to categorise her/his message, s/he needs to reflect about what s/he is going to write, and reflection is the most important skill in effective learning. Although sometimes choosing an abstraction to categorise one message can be tedious, this action helps to organise large volumes of messages and to lessens the participants' information overload [14]. Moreover, this process reduces the noise during communication, allows persistence of exchanged ideas for future reference (function of our LIM), helps to improve the resolution of conflicts and the comprehension of problems, and enhances traceability of the discussion.

3.1 A Model for the Learning Interaction Memory

In order to support interactions in learning environments, we have developed a model for the LIM. It is an extension of the IBIS argumentation model (Fig. 2) and have the following abstractions:

- Proposal - used when the participant (student or teacher) wants to introduce a theme for discussion. For example, "Let's talk about Argumentation Models".

- Argument - used to express a point of view or to explain contributions. In other words, an argument is used to express ideas, reasons for or against questions, examples and arguments. For example, "I agree with you because your explanation was clear" or "I think that using an argumentation model is interesting."
- Question - used to clarify doubts, ask someone to do something or challenge somebody to take part in a contest or to prove or justify something. For example, "Are you really sure about your answer?" or "What is an argumentation model?"
- Justification - used to clarify doubts and misunderstandings. For example, "an argumentation model provides a way to categorise the message exchanging among participants into a collaborative environment". This abstraction is used to answer a question.

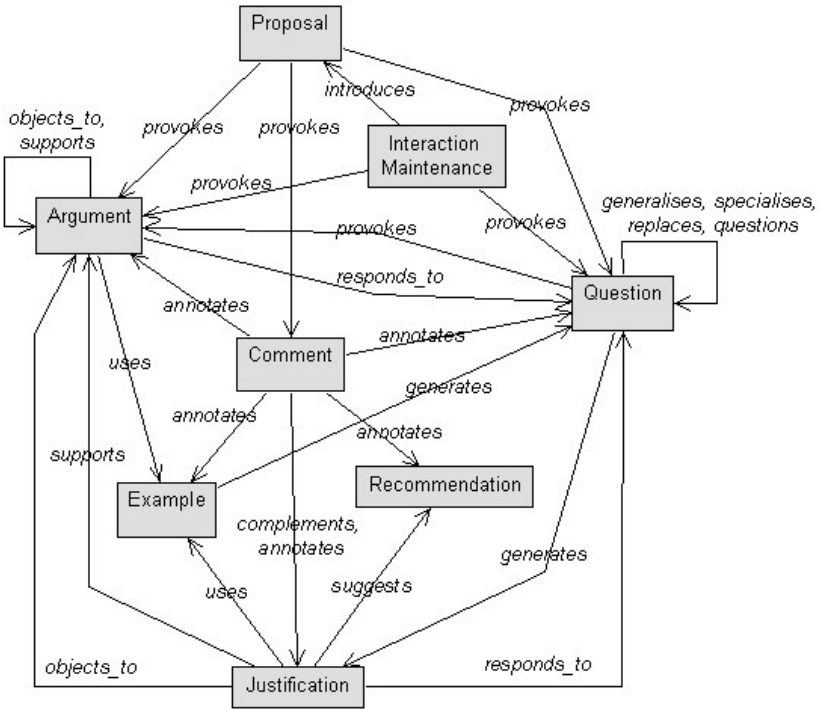


Fig. 2. A model for Learning Interaction Memory.

- Example – used to illustrate an explanation or an argument. For example, "IBIS is an example of argumentation model".
- Comment – used to complement an explanation, stimulate the student to summarize or add notes to an argument, explanation or question. In other words, comments can be used to give feedback for somebody. For example, "your explanation was very good".
- Recommendation - used to recommend references, a course of action or people to help in something. For example, "I think you should read so-and-so´s book".
- Interaction Maintenance – includes expressions used just to keep the communication channel open. For instance, "Hi, how are you?" or "Are you there?" are classified in this category.

A discussion using a learning environment is composed by all those categories. Upon sending a message, participants have to select the message category that best reflects their intention. In other words, users need to select one of a pre-defined set of appropriate utterance categories (e.g. argument, question, example, comment etc.), that provide explicit information about the content of their messages, for example:
User1 (tutor): PROPOSAL -> "Let's talk about the concept of context"
User2 (student) : COMMENT -> "It is a very interesting subject".
User3 (student): PROPOSAL -> "I propose that we talk about ontology before context"
User3 (student): ARGUMENT -> "If we talk about ontology before, it will be better to understand the idea of context first".

Consequently, the computational environment will be able to take advantage of the semantic knowledge of the categories and explicit relations among messages (e. g. if a QUESTION abstraction was used in a message, other message related to it, using our argumentation model, should use one either JUSTIFICATION, ARGUMENT or COMMENT) to organise and infer information, which will support the discussion processes. Additionally, the use of an argumentation model can help to improve conflict resolution and problem comprehension, making it easier for participants to clearly organise their ideas, better understand each other's points of view and arrive at a consensus more quickly. Once they have been categorised, messages can be stored, creating the LIM.

4. Related Works

Over the past decade, we have seen an explosion of network-based technologies that enable distant learners to work collaboratively. These CSCL environments enhance traditional distance learning curricula by giving students the opportunity to interact with other students and share ideas.
Several research projects have begun to explore the possibilities of enriching CSCL environments with tools to support collaborative interaction [21]. In many cases, interaction analysis has made use of Artificial Intelligence techniques to support their processes. However most systems have focused their analysis on quantitative and temporal elements, such as register and login of students to evidence the time of access and answer, number of students' interactions, number of correct answers and so on. More interesting approaches (e.g. [12, 24, 26, 28, 30]) have considered, in addition to these quantitative elements, qualitative factors such as the relation between a particular interaction and the past ones, as well as the analysis of the information composed by all interactions carried out during the whole learning process.
Even most modern collaborative learning systems still lack ways of supporting learner's reflection, although several works consider reflection as one of the most useful mechanisms during the learning process [19, 30, 18]. In this light, our work aims at filling in these gaps, providing ways of supporting reflection of both students and instructors, via analytical queries that can be performed on the structured interactions composing the LIM. Furthermore, using analytical queries, instructors will have access to information that supports a continuous evaluation of their students, their learning strategies and the lessons' content. Finally, the analytical queries will be a complementary tool to instructors that want more indicatives to the proper moment of interacting and motivating their students. All of these capabilities can significantly improve the potential of success of computer based learning processes.

5. Conclusions and Further Work

When an instructor decides to use Computer Supported Collaborative Learning Environments, the need for tools to evaluate and watch students taking into account interactions that occur among students and/or students and the teacher comes to light. This is due to the fact that during discussions students can think about what they did and learned, and share their knowledge with each other.
In computer-mediated education, an extremely important role of instructors is to mediate and encourage discussions. This task, however, can be troublesome when the number of participants and quantity of interactions tend to increase. So it is necessary to provide resources and information to instructors so that they can follow the evolution of students in short, medium and long terms.
In this paper, we propose a process to structure, store and query the information referring to the participants' interactions in a CSCLE. Indeed, in our analysis interactions tool, an agent society monitors the learning environment and all discussions are structured by the use of an argumentation model, which organises the types of contributions that participants make during discussion, and the way these relate to each other. A learning interaction memory is used to store the information (in a multidimensional structure). As the LIM is implemented using a data warehouse technology, analytical queries are useful to generate reports giving details of how the discussion is progressing.
Besides the analytical query mechanisms that provide several types of information to instructors, we consider that this proposal can help to evaluate the outcomes of using a tool like that in a learning environment. Teachers can also take advantage of such a tool, considering it as a support to decide on changes in their teaching strategy.
Our future research will concentrate on investigating how the concept of context [7] can be used in learning situations and formally modelling our argumentation model (using UML diagrams [6] and/or entity-relationship diagrams [15]).

References

[1] Akhras, F. N. ; Self, J. A. System Intelligence in Constructivist Learning. In International Journal of Artificial Intelligence in Education 2000, (2000) No. 11, 344-376.

[2] Araujo, R. M - QUORUM - Um Sistema de Suporte à Decisão em Grupo para o desenvolvimento de Software. M.Sc. thesis. COPPE - Universidade Federal do Rio de Janeiro, (1994).

[3] Bacelo, A. P., Becker, K. Uma ferramenta de apoio à discussão e deliberação em grupo. In: III Workshop em Sistemas Multimídia e Hipermídia (Womh'97), São Paulo, (1997). 119-130

[4] Barros, B., Verdejo, M. F. Analysing student interaction processes in order to improve collaboration. The DEGREE approach. International Journal of Artificial Intelligence in Education, (2000), n. 11,. 221-241.

[5] Bellassai, G.; Borges, M.R.; Fuller, D. A.; Pinto, J.A.; Salgado, A.C. An IBIS-based model to support group discussions. In: Proceedings of the International Workshop on the Office of the Future (IFIP). Tucson, Arizona, (1996). 39-53.

[6] Booch, G; Rumbaugh, J.; Jacobson, I. Unified Modeling Language User Guide, Hardcover, (1998).

[7] Brézillon P.; Pomerol J. Is context a kind of collective tacit knowledge? In: European CSCW 2001 Workshop on Managing Tacit Knowledge. Bonn, Germany. M. Jacovi and A. Ribak (Eds.), (2001) 23-29.

[8] Brown, A. L.; Palincsar, A. S. Guided, cooperative learning and individual knowledge acquisition. In: L.R. Resnick (Ed.), Knowing, learning and instruction. Essays in Honor of Robert Glaser. Hillsdale: Erlbaum (1989). 393-451.

[9] Cavalcanti, M.C.; Borges, M.: ARCoPAS: Um Ambiente para a Recuperação Cooperativa do Projeto Arquitetônico de Sistemas, Anais do VIII Simpósio Brasileiro de Engenharia de Software, Curitiba, PR, (1994).

[10] Cavalli-Sforza, V.; Suthers, D. D. Belvedere: An Environment for Practicing Scientific Argumentation, Notes of the AAAI Workshop on Computational Dialectics, Seattle, WA. (1994).

[11] Conklin, J, Begeman, M.L. gIBIS: A Hypertext Tool for Exploratory Policy Discussion. ACM Transactions on Office Information Systems, v. 6, no.4 , (1988).

[12] Constantino-Gonzalez, M, Suthers, D. A coached collaborative learning environment for Entity-Relationship modeling. Procedings of the 5th International Conference on Intelligent Tutoring Systems, Montreal, Canada, (2000) 324-333.

[13] CSCA (online). Computer-Supported Collaborative Argumentation Resource Site. WWW: http://kmi.open.ac.uk/people/sbs/csca. Visited in December (2003)

[14] Eleuterio M., Barthès JP, Bortolozzi F. Mediating collective discussions using an intelligent argumentation-based framework . WWW: http://newmedia.colorado.edu/cscl/168.pdf. Visited in April (2003)

[15] Elmasri, R. Navathe, S.B. Fundamentals of database systems, Califórnia,: Addison-Wesley Publishing Company, (2000).

[16] Fuks H.; Gerosa, M.A.; Lucena, C.J.P. Using the AulaNet Learning Environment to Implement Collaborative Learning via Internet. In: Innovations 2003 - World Innovations in Engineering Education and Research, iNEER, USA, (2003).

[17] Golfarelli, M.,Rizzi, S. Designing data warehouses: key steps and crucial issues. In: Journal of Computer Science and Information Management, v. 2, n..3, (1999).

[18] Goodman, B.; Soller, A.; Linton, F.; Gaimari, R. Encouraging student reflection and articulation using a learning companion. In: International Journal of Artificial Intelligence in Education, 9(3-4), (1998) 237-255.

[19] Grundy, J.; Mugridge, W.; Hosking, J.; Amor, R. Support for Collaborative, Integrated Software Development. In M. Verrall (Ed.): Proceedings of Software Engineering Environments, Noordwijkerhout, Netherlands, (1995).

[20] Inmon, W.H. Building the data warehouse (2nd ed.). New York: John Wiley & Sons, (1996).

[21] Jermann, P.; Soller, A.; Mühlenbrock, M.: From Mirroring to Guiding: a Review of the State of the Art Technology for Supporting Collaborative Learning. In: Euro-CSCL'2001, Maastrich, Holland, (2001).

[22] Kimball, R, Reeves, L, Ross, M., Thomthwaite, W. The datawarehouse lifecycle toolkit: tools and techniques for designing, developing and deploying data warehouses. New York: John Wiley & Sons, (1998).

[23] Levy, P. Les technologies de l'intelligence. L'avenir de la pensée à l'ère informatique, Ed. La découverte, (1990).

[24] Martínez, A., Dimitriadis, Y. , de la Fuente, P. Interaction Analysis for Formative Evaluation in CSCL. In: M. Llamas-Nistal, M. J. Fernández-Iglesias, and L. E. Anido-Rifón, Computer and Education: Toward a Lifelong Learning Society, Kluwer, The Netherlands, (2003).

[25] Piaget, J. Epistemologia Genética. São Paulo: Martins Fontes, (1990).

[26] Quintana, C., Eng, J., Carra, A., Wu, H., Soloway, E. Symphony: A Case Study in Extending Learner-Centered Design Through Process Space Analysis. Proceedings of CHI 99: Conference on Human Factors in Computing Systems, Pittsburgh, PA, (1999).

[27] Resta, P. Project CIRCLE: Student Mentors as a Strategy for Training and Supporting Teachers in the use of Computer-Based Tools for Collaborative Learning. Proceedings of the Computer Supported Collaborative Learning (CSCL'995). Indiana University. Bloomington, IN, (1995).

[28] Rosatelli, M. Self, J, Thirty, M. LeCS: A collaborative case study system. In: Proceedings of the 5th International Conference of Intelligent Tutoring Systems. Montreal, Canada, (2000) 242-251.

[29] Soller, A.; Wiebe, J. ; Lesgold, A. A Machine Learning Approach to Assessing Knowledge Sharing During Collaborative Learning Activities. In : Proceedings of Computer Support for Collaborative Learning 2002. Boulder, CO, (2002) 128-137.

[30] Tedesco, P. MArCo: Building an Artificial Conflict Mediator to Support Group Planning Interactions. In: International Journal of Artificial Intelligence in Education (2003), n. 13, 117-155.

[31] Wooldridge, M. Introduction to MultiAgent Systems. NY: John Wiley & Sons, (2002).

Sub-flow assignment model of multicast flows using multiple p2mp LSPs

Fernando Solano, Ramón Fabregat
Institut d'Informàtica i Aplicacions, Universitat de Girona
Av. Lluis Santaló s/n. EPS P4, (17071) Girona, España.
{fsolanod, ramon}@eia.udg.es

and

Yezid Donoso
Departamento de Ingeniería de Sistemas y Computación, Universidad del Norte
Km. 5 vía Pto. Colombia, Barranquilla, Colombia.
ydonoso@uninorte.edu.co

Abstract

In previous work, a multi-objective traffic engineering scheme (MHDB-S model) using different distribution trees to multicast several flows is proposed. Because the flow assignment can not be mapped directly into MPLS architecture, in this paper, we propose a liner system equation to create multiple point-2-multipoint LSPs based on the optimum sub-flow values obtained with our MHDB-S model.

Keywords: Multiobjective Optimization, Multicast, MPLS, Sub-flow assignment

Resumen

En trabajos previos, se ha propuesto un esquema de ingeniería de tráfico multiobjetivo (modelo MHDB-S) para realizar multicast de diversos flujos, usando diferentes árboles de distribución. Como la asignación de flujos no puede ser mapeada directamente en la arquitectura MPLS, se propone un sistema de ecuaciones lineales para crear múltiples LSPs punto-multipunto basándonos en los valores de subflujo óptimos obtenidos con nuestro modelo MHDB-S.

Palabras claves: Optimización multiobjetivo, Multicast, MPLS, Asignación de subflujos.

1.Introduction

Traffic engineering is concerned with optimizing the performance of operational networks. The main objective is to reduce congestion in hot spots and to improve resource utilization. This can be achieved by setting up explicit routes through the physical network in such way that traffic distribution is balanced across several traffic trunks [1]. Current configurations in computer networks provide an opportunity for dispersing traffic over multiple paths to decrease congestion and achieve the aggregated end-to-end bandwidth requirement.

This load balancing technique can be achieved by a multicommodity network flow formulation [2], [3] and [4], which leads to the traffic being shared over multiple routes between the ingress node and the egress nodes in order to avoid link saturation and hence the possibility of congestion. Several advantages of using multipath routing are discussed in [5]: links do not get overused and therefore do not get congested, and that multipath has the potential to aggregate bandwidth allowing a network to support more data transfer than it is possible with any one path, etc.

In previous work [6] we proposed a multi-objective traffic engineering scheme (MHDB-S model) to multicast several flows. The aim of [6] is to combine the following weighting objectives into a single aggregated metric: the maximum link utilization, the hop count, the total bandwidth consumption, and the total end-to-end delay. Moreover, our proposal solves the traffic split ratio for multiple trees.

In unicast transmission, the split ratio is fed to the routers which divide the traffic of the same pair of ingress-egress nodes into multiple paths i.e. each flow is split into multiple sub-flows. In multicast transmission, the load balancing consists of traffic being split (using the multipath approach) across multiple trees, between the ingress node and the set of egress nodes.

The proposed MHDB-S model can be applied to MPLS networks by allowing multiple explicit trees to be established in order to transport several multicast flows (fig. 1). With this load balancing technique, each flow is split between multiple trees [7] depending on the solution obtained.

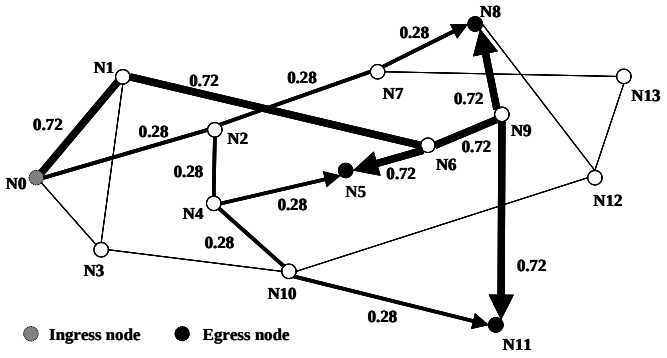


Fig. 1. Flow from ingress node N0 to egress nodes {N5, N8, N11} is split into two sub-flows, and each one is sent along different trees: {(0,1), (1,6), (6,5), (6,9), (9,8), (9,11)} and {(0,2), (2,7), (7,8), (2,4), (4,5), (4,10), (10,11)}. The sub-flow fraction along each tree is 0.72 and 0.28 respectively. Note that the total flow coming from each egress node is 1.

In this paper, we focus on the specific problem of sub-flow assignment. The aim of this is to obtain an efficient solution to formulate p2mp LSPs given a set of optimum sub-flow values.

The rest of this paper is organized as follows. In section 2, we describe some related studies. In section 3, we explain the multi-objective scheme for static multicast routing (MHDB-S model) [6] that multicast several flows and solve the traffic split ratio for multicast trees. The sub-flow assignment problem is analyzed in section 4. In section 5, we propose a linear equation system for creating multiple p2mp LSPs based on the optimum sub-flow values obtained with the MHDB-S model. The problem related to the lack of labels in MLS networks is analyzed in section 6. Finally, in section 7, we give our conclusions and suggestions for further study.

2.Related work

2.1.Multipath routing: splitting flows

Several papers, [8] [9].[10].[11] and [12], address the splitting multipath problem of unicast traffic, motivated by its importance in complete traffic engineering solutions. Traffic splitting is executed for every packet in the packet-forwarding path. A simple method to partition the input traffic is on a per-packet basis, for example in a round-robin fashion. However, this method suffers from the possibility of excessive packet reordering and is not recommended in practice.

[13] tries to balance the load among multiple LSPs according to the loading for each path. In MPLS networks [14] multiple paths can be used to forward packets belonging to the same "forwarding equivalent class" (FEC) by explicit routing. The distribution of the load in a set of alternate paths is determined by the amount of number space in a hash computation allocated to each path. Effective use of load balancing requires good traffic distribution schemes. In [15], the performance of several hashing schemes for distributing traffic over multiple links while preserving the order of packets within a flow is studied. Although hashing-based load balancing schemes have been proposed in the past, [15] is the first comprehensive study of their performance using real traffic traces.

In [8], Rost and Balakrishnan propose a multi-path transmission between sources and destinations. The current configurations in computer networks provide an opportunity for dispersing traffic over multiple paths to decrease congestion. In [8] dispersion involves (1) splitting, and (2) forwarding the resulting portions of aggregate traffic along alternate paths. The authors concentrate on (1): methods that allow a network node to subdivide aggregate traffic, and they offer a number of traffic splitting policies which divide traffic aggregates according to the desired fractions of the aggregate rate. Their methods are based on semi-consistent hashing of packets to hash regions as well as prefix-based classification.

2.2. Support of multicasting in MPLS networks

In MPLS, unicast and multicast packets have already been assigned to different type code in the link-layer header. Therefore, MPLS routers know whether a packet is from a unicast or a multicast flow. In the case of unicast forwarding the event of an incoming flow leads to the forwarding of exactly one flow. The packet duplication mechanism that is implemented in IP routers to support the IP multicast can be used to duplicate MPLS packets. MPLS routers at the bifurcation of a multicast routing tree duplicate packets and send copies of the same packet on different outgoing links. Although MPLS natively supports multicasting in its design, the MPLS community has focused its efforts mainly on the label switching of unicast IP traffic, leaving the sections on multicasting in the main MPLS documents ([14] and [16]) virtually empty, to be addressed in future studies. Based on this, there are some proposals for supporting multicasting in MPLS networks.

A framework for MPLS multicast traffic engineering proposed by Ooms et al. [17] gives an overview of the applications of MPLS techniques to IP multicast. Another proposal explains how to distribute labels for unidirectional multicast trees [18] and for bi-directional trees' label distribution [19].

To provide MPLS Traffic Engineering [18] for a point-to-multipoint (p2mp) application in an efficient manner in a large scale environment, p2mp TE mechanisms are required. Existing MPLS point-to-point (p2p) mechanisms have to be enhanced to support the p2mp TE LSP setup. [20] presents a set of requirements for p2mp TE extensions to MPLS.

In MPLS working group meeting in Seoul (march 1 2004) two different solution drafts ([21] and [22] for TE p2mp LSPs are presented but the chairs and the meeting strongly encourage the authors for both need to get together and converge on a single solution. The computation of p2mp TE paths is implementation dependent and is beyond the scope of those solutions. Path information can be computed by some off-line or on-line algorithms, e.g. the MHDB-S model presented in the next section.

[21] describes a solution for p2mp TE which extends [23] and [24] in order to establish, maintain, and teardown a p2mp TE LSP. In this case, a p2mp TE LSP is established by setting up multiple standard p2p TE LSPs from a sender node and all the downstream branch nodes along the p2p TE LSP to one of the leaf nodes of the p2mp TE LSP. The calculation for a p2mp requires three major pieces of information. The first is the route from the ingress LSR of a p2mp path to each of the egress LSRs, the second is the traffic engineering related parameters, and the third is the branch capability information.

[22] describes how RSVP-TE can be used for p2mp TE. It relies on the semantics of RSVP that RSVP-TE inherits for building a p2mp TE tree. P2p TE LSPs are set up between ingress LSR and egress LSRs. These p2p TE LSPs are appropriately merged by the network using RSVP semantics to result in a p2mp TE LSP.

Various traffic engineering solutions using programming techniques to balance loads by multiple routes have been designed and analyzed in different studies (see [6] and [25] for a detailed explanation of these proposals). It should be pointed out that several proposals can be applied to MPLS networks. In [6], we show that the multi-objective model produces a better result than various one-objective models. In [26], we present an enhanced model (MHDB-D) for multicasting dynamic groups, and in [25] and [27] we present two heuristics algorithms to solve the previous models.

2.3. The lack of labels problem.

A general problem of supporting multicasting in MPLS networks is the lack of labels. The MPLS architecture allows aggregation. Aggregation reduces the number of labels that are needed to handle a particular set of flows, and may also reduce the amount of label distribution control traffic needed [14]. Addition of new LSPs increases the label space and hence the lookup delay. So reducing the number of used labels is a desirable characteristic for any algorithm that adds LSPs to flows.

As pointed out in [14], the label based forwarding mechanism of MPLS can also be used to route along **mp2p** LSPs. In [28] and [29], aggregation algorithms that merge p2p LSPs into a minimal number of mp2p LSPs are considered. In this case, labels assigned to different incoming links are merged into one label assigned to an outgoing link. If two p2p LSPs follow the same path from an intermediate node to the egress node, these aggregation algorithms allocate the same label to the two p2p LSPs and thus reduce the number of used labels. In [30], an algorithm reducing the number of MPLS labels to N (number of nodes) + M (number of links) without increasing any link load is presented. For differentiated services with K traffic classes with different load constraints, their bound increases to K(N+M). Their stack-depth is only one, justifying implementations of MPLS with limited stack-depth.

The label stack was introduced into MPLS framework to allow multiple LSPs to be aggregated into a single LSP tunnel [14]. In [31], a comprehensive study of label size versus stack depth trade-off for MPLS routing protocols on lines and trees is undertaken. They show that in addition to LSP tunneling, label stacks can also be used to dramatically reduce the number of labels required for setting up LSPs in a network. Their protocols have numerous practical applications that include implementation of multicast trees, and virtual private networks using MPLS as the underlying signaling mechanism.

To reduce the number of used labels for multicast traffic, another label aggregation algorithm is presented in [32]. In this case, if two **p2mp** LSPs follow the same tree from an ingress node to the egress node set, the aggregation algorithm allocates the same labels to the two p2mp LSPs. Ingress nodes have a new table (named Tree Node Table) saving node information of the p2mp LSP and label allocation is executed by using this table.

Aggregated multicast is a scheme to reduce multicast state [33]. The key idea is that, instead of constructing a tree for each flow, there can be multiple multicast flows share a single aggregated tree to reduce multicast state and, hence, tree maintenance overhead and the number of used labels. Data packets from different flows are multiplexed in the same distribution tree, called aggregated tree. Each data packet of each group is encapsulated and travels through the aggregated tree.

3. Optimization model

The network is modeled as a directed graph $G=(N,E)$, where *N* is the set of nodes and *E* is the set of links. The set of links is $E \subseteq N \times N$. We use *n* to denote the number of network nodes, i.e., $n=|N|$. Among the nodes, we have a source $s \in N$ (ingress node) and some destinations T (the set of egress nodes). Let be $t \in T$ any egress node. Let $(i,j) \in E$ be the link from node *i* to node *j*. Let $f \in F$ be any multicast flow, where *F* is the flow set and T_f is the egress node subset to the multicast flow *f*. We use |*F*| to denote the number of flows. Note that $T = \cup T_f$.

Let X_{ij}^{tf} be the fraction of flow *f* to egress node *t* assigned to link *(i,j)*; note that these variables include the egress node *t*. Including the egress node variables allows us to control the bandwidth consumption in each link with the destination of the set of egress nodes. Therefore, it is possible to maintain the constraint of flow equilibrium to the intermediate nodes exactly. The problem solution, X_{ij}^{tf} variables, provides optimum flow values.

Let c_{ij} be the capacity of each link *(i,j)*. Let bw_f be the traffic demand of a flow *f* from the ingress node *s* to T_f. The binary variables Y_{ij}^{tf} represent whether link *(i,j)* is used (1) or not (0) for the multicast tree rooted at the ingress node *s* and reaching the egress node subset T_f. Let v_{ij} be the propagation delay of link *(i,j)*. Let *m* be the number of variables in the multi-objective function. Let $connection_{ij}$ be the indicator of whether there is a link between nodes *i* and *j*.

The problem of minimizing |*F*| multicast flows from ingress node *s* to the egress nodes of each subset T_f is formulated as follows:

Minimize

$$r_1.\alpha + r_2 \sum_{f\in F}\sum_{t\in T_f}\sum_{(i,j)\in E} Y_{ij}^{tf} + r_3 \sum_{f\in F}\sum_{(i,j)\in E} bw_f \max_{t\in T_f}\left(X_{ij}^{tf}\right) + r_4 \sum_{f\in F}\sum_{t\in T_f}\sum_{(i,j)\in E} v_{ij} Y_{ij}^{tf} \quad \text{(MHDB-S model)} \quad (1)$$

Subject to

$$\sum_{(i,j)\in E} X_{ij}^{tf} - \sum_{(j,i)\in E} X_{ji}^{tf} = 1 \quad ,t\in T_f, f\in F, i=s \quad (2)$$

$$\sum_{(i,j)\in E} X_{ij}^{tf} - \sum_{(j,i)\in E} X_{ji}^{tf} = -1 \quad ,i,t\in T_f, f\in F \quad (3)$$

$$\sum_{(i,j)\in E} X_{ij}^{tf} - \sum_{(j,i)\in E} X_{ji}^{tf} = 0 \quad ,t\in T_f, f\in F, i\neq s, i\notin T_f \quad (4)$$

$$\sum_{f\in F} bw_f.\max_{t\in T_f}(X_{ij}^{tf}) \le c_{ij}.\alpha \quad ,\alpha\ge 0,(i,j)\in E \quad (5)$$

$$\sum_{j\in N}\sum_{t\in T_f} Y_{ij}^{tf} \le \left\lceil \frac{bw_f}{\left\lfloor \sum_{j\in N} c_{ij} \Big/ \sum_{j\in N} connection_{ij} \right\rfloor} \right\rceil \quad ,i\in N, f\in F \quad (6)$$

where

$$X_{ij}^{tf} \in \Re, 0 \le X_{ij}^{tf} \le 1 \quad (7)$$

$$Y_{ij}^{tf} = \left\lceil X_{ij}^{tf} \right\rceil = \begin{cases} 0 & ,X_{ij}^{tf} = 0 \\ 1 & ,0 < X_{ij}^{tf} \le 1 \end{cases} \quad (8)$$

$$\sum_{i=1}^{m} r_i = 1, \quad r_i \in \Re, \quad r_i \ge 0, m > 0 \quad (9)$$

The Multi-objective function (MHDB model) (1) defines a function and generates a single aggregated metric through a combination of weighting objectives. The main objective consists in minimizing the maximum link utilization (MLU), which is represented as α in equation (1). In this case, the solution obtained may report long routes. In order to eliminate these routes and to minimize hop count (HC), the term $\sum_{f\in F}\sum_{t\in T_f}\sum_{(i,j)\in E} Y_{ij}^{tf}$ is added. In order to minimize the total bandwidth consumption (BC) over all links, the term $\sum_{f\in F}\sum_{t(i,j)\in E} bw_f \max_{t\in T_f}\left(X_{ij}^{tf}\right)$ is also added. This is included so that, if there is more than one solution with the best maximum link utilization, the solution with the minimum resource utilization is chosen. Though several sub-flows of the flow *f* in the link *(i,j)* with destinations to different egress nodes are sent, in multicast IP specification just one sub-flow will be sent, that is, only the maximum value of X_{ij}^{tf} for $t\in T_f$ needs to be considered. Furthermore, in order to minimize the total end-to-end propagation delay (DL) over all the links, the term $\sum_{f\in F}\sum_{t\in T_f}\sum_{(i,j)\in E} v_{ij}\, Y_{ij}^{tf}$ is also added.

Constraints (2), (3) and (4) are flow conservation constraints. Constraint (2) ensures that the total flow emerging from ingress node to any egress node *t* at flow *f* is 1. Constraint (3) ensures that the total flow coming from an egress node *t* at flow *f* is 1. Constraint (4) ensures that for any intermediate node different from the ingress node (i ≠ s) and egress nodes $(i\notin T)$, the sum of their output flows to the egress node *t* minus the input flows with destination egress node *t* at flow *f* is 0.

Constraint (5) is the maximum link utilization constraint. In a unicast connection, the total amount of bandwidth consumed by all the flows with the destination of egress node *t* must not exceed the maximum utilization (α) per link capacity c_{ij}, that is, $\sum_{f\in F} bw_f \sum_{t\in T} X_{ij}^{tf} \le c_{ij}\,\mathrm{A},(i,j)\in E$. Nevertheless, in constraint (5) only the maximum value of X_{ij}^{tf} for $t\in T_f$ needs to be considered.

Constraint (6) limits the maximum number of sub-flows (MSF) in each node by means of the capacity of each link and the traffic demand. This formulation represents the amount of necessary links for a particular traffic demand. Without this constraint, the model could suffer from scalability problems, i.e. the label space used by LSPs would be too high.

Expression (7) shows that the X_{ij}^{tf} variables must be real numbers between 0 and 1. These variables form multiple trees transport multicast flow. The demand between the ingress node and the egress node *t* may be split over multiple routes. When the problem is solved without load balancing, this variable will only be able to take values 0 and 1, which will show, respectively, whether or not the link *(i,j)* is used to carry information to egress node *t*.

Expression (8) calculates Y_{ij}^{tf} as a function of X_{ij}^{tf} .

Finally, expression (9) shows that the weighting coefficients, r_i, assigned to the objectives are normalized. These values are calculated by solving the optimization problem.

4. Sub-flow Assignment to p2mp LSPs problem

In this section, we detail the problem of creating multiple p2mp LSPs based on the optimum sub-flow values X_{ij}^{tf} obtained with solutions to MHDB-S model (1). Remember that X_{ij}^{tf} is the fraction of flow *f* to destination node *t* assigned to link *(i,j)*. First at all, because the following system applies only to one flow *f*, the index *f* will be omitted when it does not cause confusions. It means that, for example, X_{ij}^{t} is the subset of links in X_{ij}^{tf} that transmits the flow *f*.

To explain the problem, the MHDB-S model have been applied to the topology of fig. 2, with a single flow *f*, where s=N1 and T={ N5, N6}. In this case, a possible sub-flow solution (X_{ij}^{t}) obtained is shown in fig. 3. The simplest solution (fig. 4), to create LSPs based on the optimum sub-flow values, is to send each sub-flow (0,4 and 0,6 fraction) to the group separately, and in this case each sub-flow is assigned to one p2mp LPS. In fig. 4, each packet represents a 0,2 fraction of the flow. With this assignment, sub-flows X_{12}^{5} and X_{12}^{6} are different and the maximum link utilization constraint (5) could be violated. Moreover, the network is inefficiently used because multicast node capabilities are not considered. Only ingress node multicast capabilities are considered when applying the multipath approach, which permits that the flow is balanced across several links.

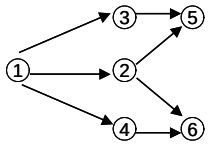


Fig. 2. Physical network topology.

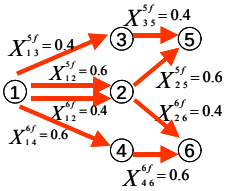


Fig. 3. MDDB-S solution

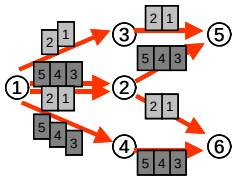


Fig. 4. Simplest p2mp assignment.

A second approach considers that one sub-flow is included in the other, i.e. $\min(X_{12}^{5}, X_{12}^{6}) \subseteq \max(X_{12}^{5}, X_{12}^{6})$, in the example $X_{12}^{5} \subseteq X_{12}^{6}$. If both sub-flows X_{12}^{5} and X_{12}^{6} are sent over the link (1,2) to each member of the group separately (fig. 5), a part of the same flow is being transmitted over the same link and the network is also inefficiently used.

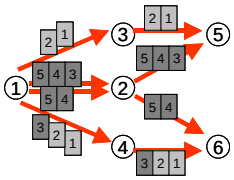


Fig. 5. p2mp assignment: unicast transmission

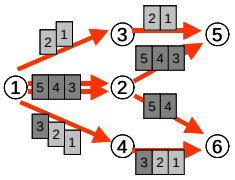


Fig. 6. p2mp assignment: multicast transmission

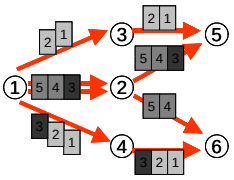


Fig. 7. p2mp assignment: sub-flow assignment

Moreover, if a node has multicast capabilities, it is not necessary to transmit all the sub-flows over the link. In particular, if N2 has multicast capabilities, only the max (X_{12}^5 , X_{12}^6) must be transmitted over link (1,2) (fig. 6). However, this solution presents a problem in the forwarding mechanism. Some incoming packets at node 2 must be forwarded exactly once (packet 3), but other packets of the same sub-flow must be forwarded by different output links (packets 4 and 5). To solve this, the ingress node must split this sub-flow in several sub-flows (fig. 7).

5. Sub-flow assignment based on a linear equation system

In this section, a linear system equation to split a sub-flow in several sub-flows is proposed. Solution is the set of desired multicast trees for the set of X_{ij}^{tf} values. For a unique flow *f* and given values X_{ij}^t , the δ solution of the presented system is a set of encoded *p2mp* LSPs, in which each, $p2mp_k$, sends a fixed fraction, c_{min} , of the whole bandwidth to all destinations in T_f . Let $\delta_{ij}^{p2mp_k}$ be a natural number, possibly 0, indicating the number of destinations that link (i,j) at $p2mp_k$ broadcasts.

To compute the c_{min} value, it should be taken into consideration that a low value can result in many equation with many equal p2mp LSPs and, in contrast, a high value can result in a unsolvable problem because some fractions of X_{ij}^t could not be assigned to p2mp LSP. In the other hand, it can be seen that the set of found p2mp LSPs, and thus δ can be regarded as a linear combination of X_{ij}^t values. Therefore, an optimal c_{min} value must divide all X_{ij}^t and difference among them, hence $c_{min}=m.c.d.^*(X_{ij}^t)$, where $m.c.d.^*$ is the maximum common divisor operator used with real numbers between 0 and 1. In the example of figure 3, the c_{min} value is 20%.

The following three equation sets model p2mp LSPs in general.

$$\forall k, \forall j \quad \sum_{j \in N} \delta_{sj}^{p2mp_k} = |T_f| \tag{10}$$

$$\forall k, \forall (i,j) \in E, \forall m \quad \delta_{ij}^{p2mp_k} = \begin{Bmatrix} \sum_{m \in N} \delta_{jm}^{p2mp_k} + 1, & if \ j \in T_f \\ \sum_{m \in N} \delta_{jm}^{p2mp_k}, & otherwise \end{Bmatrix} \tag{11}$$

$$\forall k, \forall t \in T_f \quad \sum_{i \in N} \delta_{it}^{p2mp_k} = 1 \tag{12}$$

And for this problem in particular:

$$\forall (i,j) \in E \quad c_{min} \sum_k \delta_{ij}^{p2mp_k} = \sum_{t \in T_f} X_{ij}^t \tag{13}$$

Note that, the set $\{(i,j) | \delta_{ij}^{p2mp_k} \geq 1\}$ for a given *k* is the set of links that conforms the tree $p2mp_k$.

The equation set (10) suggests that the number of reached destinations from a source node *s* is equal to $|T_f|$. This is clear because all p2mp LSPs reach exactly all destinations. Another obvious consequence is that $|p2mp|=1/c_{min}$, in other words, the number of constructed p2mp LSPs is the inverse of the fraction sent by each p2mp LSP. For the analyzed example at figure 3, the number of p2mp LSPs to be constructed is 5.

The conservation flow law seen at (2), (3) and (4) can be traduced as the set regarded on (11). It means that the amount of destinations a node *j* must forward packets to is the same amount after (i.e. *i* node) and before (i.e. *m* nodes), or one less if j is a destination.

Inside a p2mp LSP, no destination node can receive flow information by two links at the same time. This can be expressed with (12) where *t* is a leaf node in the p2mp LSPs, i.e. it does not reroute traffic to another point in the network.

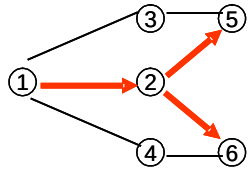
By looking at a single link *(i,j)*, the total bandwidth consumed in a link for a single destination should be equal to the amount consumed by all p2mp LSPs to that destination. In general, this holds for all destinations at the same

time by (13).

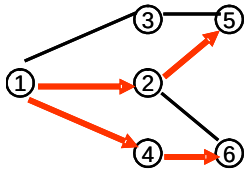
For a given set of x_{ij}^{t} values, the δ solution set of the presented system is a set of encoded p2mp LSPs, which are conformed by those δ values greater or equal to 1. Note that the equation system can resolve into equal p2mp LSPs. If two p2mp LSPs A and B are the same (i.e. contains the same links, no more, no less), these p2mp LSPs can be merged by adding their fractions. Solving the example described on figure 3, the solution (in table 1) shows that 2 pairs of p2mp LSPs (1 $\wedge$ 4 and 3 $\wedge$ 5) can be merged. Therefore, we have a set of 3 p2mp LSPs in our solution (figure 8); one transmitting 20% of the total bandwidth and two transmitting 40% of it.

$\{\delta_{12}^{1\wedge4}=2, \delta_{25}^{1\wedge4}=1, \delta_{26}^{1\wedge4}=1\}$	$\{\delta_{12}^{2}=1, \delta_{25}^{2}=1, \delta_{14}^{2}=1, \delta_{46}^{2}=1\}$	$\{\delta_{13}^{3\wedge5}=1, \delta_{35}^{3\wedge5}=1, \delta_{14}^{3\wedge5}=1,, \delta_{46}^{3\wedge5}=1\}$

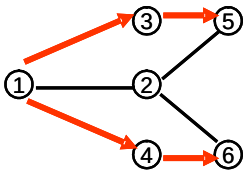
Table 1: δ solution. Super indexes $a\wedge b$ indicates that the values were assigned initially to two p2mp LSPs (a and b).



a. First tree. (40%)



b. Second tree. (20%)



c. Third tree. (40%)

Fig. 8. The p2mp LSPs.

We analyze the performance of the sub-flow assignment algorithm when sub-flow assignment and merging LSPs is considered. Over the 14-node NSF (National Science Foundation) network (fig. 1), two flows with the same source, s=N0, are transmitted. The egress nodes subsets are T1={N5, N8, N11} and T2={N8, N11, N13} respectively. The transmission rates are 256 Kbps, 512 Kbps, 1 Mbps, 1.5 Mbps, 2 Mbps and 2.5 Mbps for each flow.

<table>
<tr><th colspan="2"></th><th>0.25</th><th>0.5</th><th>1</th><th>1.5</th><th>2</th><th>2.5</th></tr>
<tr><td>A</td><td>MHDB-S model:
(multicast + multipath)</td><td colspan="6">p2mp LSPs are not considered (see section 4)</td></tr>
<tr><td>B</td><td>MHDB-S +
Sub-flow assignment</td><td colspan="3">c_{min} = 33 %
f1 = 3
f2 = 3</td><td colspan="3">c_{min} = 10 %
f1=10
f2=10</td></tr>
<tr><td>C</td><td>MHDB-S +
Sub-flow assignment +
merging LSPs</td><td colspan="3">f1 = 2
f2 = 2</td><td>f1 = 8
f2 = 5</td><td colspan="2">f1 = 9
f2 = 6</td></tr>
<tr><td colspan="8">Table 2: Total number of p2mp LPS obtained to flow 1 and flow 2</td></tr>
</table>

Table 2 shows the number of p2mp LSPs used when different schemes for multicasting several flows are considered. Comparing B and C, the merging LSPs scheme reduces the number of p2mp LSPs.

7. Conclusions

In this paper, considering a multiobjective traffic engineering scheme using different distribution trees to multicast several multicast flows, we propose a sub-flow assignation method based on a linear equation system to create multiple p2mp LSPs. The comparison of different routing schemes shows that the number of LSPs found with the sub-flow assignation method is more than the number of flows considered. However, the merging LSPs scheme reduces these values.

In the future we plan to demonstrate the usefulness of sub-flow assignment to p2mp LSPs with a variety of network scenarios. Despite merging LSPs reduces the number of used label, label aggregation algorithms will be also considered because they reduce even more the labels needed. Moreover, label stacking mechanism will be also analyzed to reduce more this value.

Acknowledgments

This work was partially supported by the MCyT under the project TIC2003-05567. The work of Yezid Donoso was supported by the Universidad del Norte (Colombia) under contract G01 02-MP03-CCBFPD-0001-2001. The work of Fernando Solano was supported by the Ministry of Universities, Research and Information Society (DURSI) of the Government of Catalonia under contract 2004FI-00693.

References

[1] Changhoon Kim, Yanghee Choi, Yongho Seok, Youngseok Lee. "A constrained Multipath Traffic Engineering Scheme for MPLS Networks". ICC 2002.
[2] R. Ahuja, T. Magnanti, J. Orlin. "Network flows: Theory, algorithms and applications". Prentice-Hall NJ USA 1993.
[3] M. Bazaraa, J. Jarvis, H. Sherali. "Linear Programming and Network Flows". John Wiley & Sons, 2nd ed, USA, 1990.
[4] M. Bazaraa, H. Sherali, C. M. Shetty. "Nonlinear Programming, Theory and Algorithms". John Wiley & Sons, 2nd ed, USA, 1993.
[5] J.C. Chen, S.H. Chan. "Multipath routing for video unicast over bandwidth-limited networks". GLOBECOM 2001
[6] Y. Donoso, R. Fabregat, L. Fàbrega. "Multi-Objective scheme over multi-tree routing in multicast MPLS networks". ACM/IFIP LANC'03.
[7] Y. Wang, Z. Wang, L. Zhang. "Internet Traffic Engineering without Full Mesh Overlaying". INFOCOM 2001.
[8] S. Rost, H. Balakrishnan. "Rate-Aware Splitting of Aggregate Traffic". Submitted http://web mit edu/stanrost/www/research/publications html
[9] C. Cetinkaya, E. Knightly. "Opportunistic Traffic Scheduling Over Multiple Network Paths". INFOCOM 2004.
[10] A. Sridharan; R. Guerin, C. Diot. "Achieving near-optimal traffic engineering solutions for current OSPF/IS-IS networks". INFOCOM 2003.
[11] S. Vutukury, J. J. Garcia-Luna-Aceves. "A traffic engineering approach based on minimum-delay routing". Ninth International Conference on Computer Communications and Networks, October 2000, pp 42 - 47.
[12] H. Cho, J. Lee, B. Kim. "Multi-path Constraint-based Routing Algorithms for MPLS Traffic Engineering". ICC '03. Volume: 3 pp 1963 - 1967.
[13] C. Villamizar. "MPLS optimized multipath (MPLS-OMP)". Internet draft. 1998.
[14] E. Rosen, A. Viswanathan, R. Callon. "Multiprotocol Label Switching Architecture". RFC 3031. January 2001.
[15] Z. Cao, Z. Wang, E. Zegura. "Performance of hashing-based schemes for Internet load balancing". INFOCOM 2000.
[16] L. Andersson, P. Doolan, N. Feldman, A. Fredette, B. Thomas, "LDP Specification". RFC 3036. January 2001.
[17] D. Ooms, B. Sales, W. Livens, A. Acharya, F. Griffoul, F. Ansari. "Overview of IP Multicast in a Multi-Protocol Label" RFC 3353. August 2002.
[18] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, J. McManus. "Requirements for Traffic Engineering over MPLS". RFC 2702. September 1999.
[19] J. Cui, J. Kim, A. Fei, M. Faloutsos, M. Gerla. "Scalable QoS multicast provisioning in Diff-Serv-Supported MPLS networks". GLOBECOM 2002.
[20] S. Yasukawa, D. Papadimitriou, J. Ph. Vasseur, Y. Kamite, R. Aggarwal, A. Kullberg. "Requirements for point-to-multipoint extension to RSVP-TE". draft-yasukawa-mpls-p2mp-requirement-02.txt. March 2004.
[21] S. Yasukawa, A. Kullberg, L. Berger. "Extended RSVP-TE for point-to-multipoint LSP tunnels. Draft-yasukawa-mpls-rsvp-p2mp-04.txt. February 2004.
[22] R. Aggarwal, L. Wei, G. Apostolopoulos, K. Kompella, J. Drake. "Establishing point to multipoint MPLS TE LSPs". draft-raggarwa-mpls-p2mp-te-02.txt. January 2004.
[23] D. Awduche , L. Berger, D. Gan, T. Li, V. Srinivasan, G. Swallow. "RSVP-TE: Extensions to RSVP for LSP Tunnels". RFC 3209. December 2001
[24] L. Berger. "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions". January 2003
[25] Y. Donoso, R. Fabregat and J. L. Marzo. "Multi-objective optimization algorithm for multicast routing with traffic engineering". IEEE ICN'04.
[26] Y. Donoso, R. Fabregat. J. L. Marzo. "Multi-Objective optimization scheme for dynamic multicast groups". IEEE ISCC'04.
[27] Y. Donoso, R. Fabregat, J. L. Marzo. "Multi-Objective Optimization Model and Heuristic Algorithm for

Dynamic Multicast Routing". IEEE & VDE Networks 2004.
[28] H. Saito, Y. Miyao, M. Yoshida. "Traffic engineering using multiple multipoint-to-point LSPs". INFOCOM 2000.
[29] S. Bhatnagar, S. Ganguly, B. Nath. "Label space reduction in multipoint-to-point LSPs for traffic engineering". ECUMN2002.
[30] D. Applegate, M. Thorup. "Load optimal MPLS routing with N+M labels". INFOCOM'03.
[31] A. Gupta, A. Kumar, R. Rastogi. "Exploring the trade-off between label size and stack depth in MPLS Routing. INFOCOM'03.
[32] Y.K. Oh, D.K. Kim, H.J. Yoen, M.S. Do, J. Lee. "Scalable MPLS multicast using label aggregation in Internet broadcasting systems". ICT'03.
[33] A. Fei, J.H. Cui, M. Gerla, M. Faloutsos. "Aggregated multicast: an approach to reduce multicast state". Sixth Globel Internet Symposium (GI2001), Nov. 2001.

Uma Hieraquia para Classificação de Protocolos Otimistas de Sincronização em Simulação Distribuída

Renata S. Lobato
Dep. de Ciências de Computação e Estatística, IBILCE, UNESP
Rua Cristóvão Colombo, 2265
São José do Rio Preto, Brasil, 15054-000
renata@dcce.ibilce.unesp.br

and

Marcos J. Santana e Regina H. C. Santana
Dep. de Ciências de Computação e Estatística, ICMC, USP
Av. do Trabalhador São-Carlense, 400 Caixa Postal 668
São Carlos, Brasil, 13560-970
{mjs, rcs}@icmc.usp.br

and

Roberta S. Ulson
Dep. de Computação, Faculdade de Ciências, UNESP
Av. Engenheiro Luis Edmundo Carrijo Coube, s/n, Caixa Postal 473
Bauru, Brasil, 17033-360
roberta@fc.unesp.br

Abstract

This paper presents an updated classification for optimistic distributed synchronization simulation protocols. As the distributed simulation performance depends on two main factors, events synchronization costs (the protocol that is used) and process communication costs, the taxonomy can help researchers and distributed simulation users to study these aspects, by grouping the protocols with similar characteristics.

Keywords: Distributed Simulation, Synchronization Protocols

Resumo

Este artigo apresenta uma versão estendida da classificação para protocolos de simulação distribuída otimista. Como o desempenho da simulação distribuída depende de dois fatores principais, o custo da sincronização de eventos (o protocolo utilizado) e o custo da comunicação entre processos, a classificação pode auxiliar pesquisadores no estudo desses aspectos, estando agrupados os protocolos com características semelhantes

Palavras chaves: Simulação Distribuída, Protocolos de Sincronização

1 INTRODUCÃO

O protocolo conservativo *CMB* (Chandy, Misra e Bryant) [14] e o protocolo otimista *Time Warp* [10] são dois protocolos de sincronização em simulação distribuída discutidos amplamente na literatura nos últimos 20 anos. Não existe um consenso sobre qual dos dois apresenta melhor desempenho e por esse motivo tem-se optado por desenvolver protocolos que buscam explorar as vantagens que os protocolos *CMB* e *Time Warp* apresentam. Assim, podem-se inserir características conservativas em um protocolo puramente otimista, ou então tornar mais otimista um protocolo conservativo. A primeira opção, por exemplo, deu origem a muitas variantes do protocolo otimista *Time Warp*.

Um novo problema surgiu para aqueles que procuravam uma alternativa entre *CMB* e *Time Warp*, envolvendo a tomada de decisão sobre qual dos protocolos otimistas utilizar para a resolução de problemas.

Uma avaliação entre esses protocolos fica facilitada quando se consegue organizá-los, agrupando os protocolos que possuem características semelhantes. Com uma classificação desse tipo, pode-se avaliar um grupo ao invés de um protocolo específico.

As classificações apresentadas na literatura são redundantes e confusas, o que levou à proposta de uma nova taxonomia para facilitar o estudo dos protocolos otimistas que inserem noções de conservadorismo no *Time Warp* [12]. Essa classificação é expandida neste artigo.

Este artigo é organizado da seguinte forma: a seção 2 faz uma revisão sobre classificações de protocolos otimistas, a seção 3 estende as classificações já publicadas pelo autor [25] [12], e a seção 4 mostra as conclusões do trabalho.

2 REVISÃO E CRÍTICA DE CLASSIFICAÇÕES EXISTENTES

As pesquisas na área de simulação distribuída têm convergido para variações híbridas de protocolos de sincronização que implementam uma abordagem intermediária entre os extremos conservativo e otimista, uma vez que a escolha por um determinado protocolo é uma tarefa bastante complexa. Esses protocolos utilizam uma forma de otimismo controlado, aproveitando os benefícios da visão otimista (explorar mais paralelismo) sem serem afetados pelos seus problemas (sobrecarga causada por *rollbacks* e consumo de memória), ou adicionam mais otimismo em alguma abordagem conservativa.

Na literatura de simulação distribuída estão descritos vários protocolos para sincronização que são basicamente variantes dos protocolos *CMB* e *Time Warp*. A quantidade de protocolos leva à necessidade de uma classificação que permita estudá-los de uma forma mais organizada. A construção de uma classificação concisa e hierárquica pode facilitar as análises comparativas de desempenho entre os protocolos. O uso da classificação pode identificar os aspectos mais importantes do protocolo *Time Warp* e que merecem atenção em um estudo de desempenho através da técnica de modelagem.

2.1 Classificação de Das

A classificação apresentada por Das [5] divide os protocolos em Híbridos ou Adaptativos. Os protocolos Híbridos abrangem os protocolos que adicionam otimismo a protocolos conservativos, como por exemplo Simulação Especulativa e SRADS [5], *Breathing Time Buckets* [29], *Filtered Rollback* [5], e os protocolos que buscam controlar o otimismo no *Time Warp*, como por exemplo *Moving Time Windows* [28], *Bounded Time Warp* [5], MIMDIX [20], *Breathing Time Warp* [30], *Composite ELSA* [5], *Local Time Warp* [20] e *Filter* [16].

Os protocolos Adaptativos ajustam-se entre a abordagem otimista e a abordagem conservativa, com o objetivo de minimizar o tempo de execução da simulação distribuída. São subdivididos em Gerais, Estado Local e Estado Global. Os protocolos Gerais têm como exemplo *Moving Time Windows* e MIMDIX. Os protocolos com base em informações referentes ao Estado Local do processo lógico englobam *Penalty-based Throttling* [22], *Adaptive Time Warp* [7], Protocolo Probabilístico baseado em Custos [5], *Probabilistic Distributed Discrete Event Simulation Protocol* [8], *Local Adaptive Protocol* [5], *Probabilistic Direct Optimism Control* (PADOC) [7] e *Adaptive Bounded Time Window* [5]. Os protocolos que utilizam informações referentes ao Estado Global da simulação distribuída agrupam *Adaptive Memory Management Protocol*, *Near Perfect State Information* (NPSI) e *Cancelback Protocol* [7].

2.2 Classificação de Srinivasan

A classificação e a classe de protocolos propostos por Srinivasan foram publicadas em trabalhos distintos e abordam de forma independente os protocolos que limitam o otimismo no Time Warp [27] e os protocolos que se adaptam às mudanças no estado da simulação [26]. Os protocolos são divididos em: Janelas, Espaço, Penalidade, Conhecimento, Probabilísticos e Estado Global.

Os protocolos com base em Janelas permitem que somente os eventos com valor de marca de tempo dentro do intervalo da janela de tempo sejam executados. São exemplos *Moving Time Windows*, *Window based Throttling* [22], *Bounded Time Warp* e *Breathing Time Warp*.

Os protocolos baseados no Espaço utilizam limites espaciais ao invés de limites temporais para limitar o otimismo. Os processos lógicos são divididos em grupos, e *Time Warp* é utilizado para efetuar o sincronismo dentro de cada grupo. Os grupos interagem de forma conservativa. Pode-se citar como exemplo *Local Time Warp* e *SRADS*.

De acordo com o autor, nos protocolos baseados em Penalidade utiliza-se o comportamento da simulação para penalizar (bloquear) processos lógicos enquanto outros são favorecidos (podem continuar a execução). Como exemplo tem-se *Penalty-based Throttling*.

Os protocolos com base em Conhecimento utilizam informação sobre a ocorrência de *rollbacks* para restringir a propagação de computação provavelmente incorreta. Exemplos incluem *Filter* e WOLF [20].

Os protocolos Probabilísticos fazem uma previsão probabilística sobre o comportamento dos processos lógicos. Como exemplo tem-se MIMDIX.

Os protocolos com base no Estado Global da simulação analisam a simulação distribuída como um todo para limitar o excesso de otimismo do *Time Warp*. Exemplos são *Adaptive Memory Management* e os protocolos NPSI.

Srinivasan [26] também propõe que o controle do otimismo no *Time Warp* seja classificado, devido ao fato de que alguns protocolos introduzem atrasos entre as execuções dos eventos. Esses protocolos foram denominados *Asynchronous Adaptive Waiting Protocols* (AAWP). Como exemplos pode-se citar *Penalty-based Throttling*, *Adaptive Time Warp*, *Local Adaptive Protocol*, protocolos NPSI, *Breathing Time Warp* e UDS.

3 CLASSIFICAÇÃO PARA PROTOCOLOS OTIMISTAS EM SIMULAÇÃO DISTRIBUÍDA

Este trabalho sugere uma extensão para as classificações de Das e Srinivasan, buscando agrupar de forma natural a grande variedade de protocolos propostos ao longo dos anos [25] [12]. Essa classificação permite uma melhor visualização dos protocolos otimistas, consistindo em uma abordagem genérica e hierárquica.

Além disso, nenhuma das classificações apresentadas considera o fato de que existem diferentes algoritmos que podem ser seguidos para efetuar determinadas tarefas, como por exemplo o salvamento de estados e o cancelamento de mensagens. Esses e outros algoritmos, como mostrado na literatura, podem influenciar no desempenho do protocolo e, por isso, serão considerados na classificação apresentada nesta seção.

A classificação de Das apresenta-se inconsistente ao separar os Protocolos que Controlam Otimismo dos Protocolos Adaptativos. Ambos procuram limitar o otimismo do *Time Warp*, podendo executar este trabalho de forma estática (não adaptativa) ou dinâmica (adaptativa), que Das não menciona. Além disso, a noção de protocolo Híbrido abrange também aqueles que adicionam otimismo a protocolos conservativos, ou seja, protocolos cujas características principais são as da abordagem *CMB*. Esses protocolos, por apresentarem características intrínsecas de abordagens conservativas, como por exemplo o uso de mensagens nulas, não são considerados [12].

Das também classifica *Local Adaptive Protocol* [5] como um protocolo que se utiliza do estado da simulação para limitar o otimismo. Apesar de cumprir esse papel, esse protocolo apresenta *deadlocks* e mensagens nulas, como no *CMB*, e também não será considerado na taxonomia proposta. A classificação de Srinivasan cria grupos de protocolos com base em Penalidade e Estado Global e não considera protocolos que fazem uso de informações referentes ao estado local do processo lógico. A característica básica de todos esses protocolos de sincronização é a utilização de informações relativas ao estado da simulação para definir se os processos lógicos podem ou não executar eventos, ou seja, têm como base o Estado da simulação. O grupo dos protocolos com base no Conhecimento também utiliza informação relativa ao estado da simulação distribuída, porém essa informação é difundida para outros processos lógicos. Nesse caso, a denominação mais adequada é Difusão. Srinivasan também não separa os protocolos adaptativos dos não adaptativos.

Além disso, ambas as classificações tratam o protocolo *Adaptive Memory Management* como sendo um protocolo baseado no estado global da simulação. É correto afirmar que esse protocolo, e também o protocolo *Cancelback*, limitam o otimismo do *Time Warp* observando o estado da simulação, porém essa limitação é feita de maneira indireta. O objetivo principal consiste em efetuar o gerenciamento da memória disponível para salvar estados, mas ao imporem aos processos lógicos um espaço limitado na memória disponível estão também prevenindo que os mesmos avancem muito no tempo de simulação. Isso justifica a criação de uma nova classe na taxonomia, como explicado nas seções seguintes, como apresentado em [12].

A classificação para os protocolos otimistas foi desenvolvida em uma estrutura concisa que fornece uma visão global dos protocolos. Essa classificação é organizada da seguinte forma: de acordo com as Características de Implementação do protocolo de sincronização (figura 1), de acordo com as Características do Protocolo de Sincronização propriamente dito (figura 2) [25] e de acordo com o Modo de Limitar o Otimismo (figura 3).

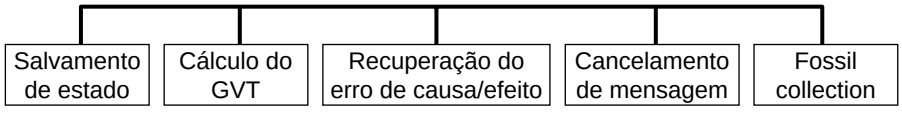


Figura 1: Organização de Protocolos segundo as Características de Implementação

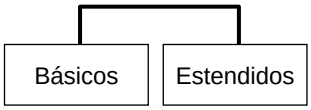


Figura 2: Organização de Protocolos segundo as Características do Protocolo de Sincronização

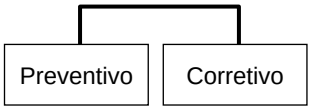


Figura 3: Organização de Protocolos segundo o Modo de Limitar o Otimismo

Essa classificação considera apenas os protocolos que têm como base a limitação do otimismo no *Time Warp*, o que exclui protocolos como, por exemplo, *Local Adaptive Protocol*, que está incluído nas classificações de Das e Srinavasan, e os protocolos que adicionam otimismo a protocolos conservativos classificados por Das.

A organização que considera as características de Implementação é discutida na seção 3.1, estando justificada pelo fato de que não existe um consenso na área de simulação distribuída com relação à melhor maneira de se implementar protocolos baseados no Time Warp [1].

A organização com base no Protocolo de Sincronização apresenta os protocolos divididos em duas categorias, Protocolo Básico e Protocolos Estendidos. O primeiro corresponde ao mecanismo proposto por Jefferson [10] enquanto os Protocolos Estendidos correspondem a variantes do *Time Warp* que procuram reduzir o excesso de otimismo da idéia original de Jefferson. A seção 3.2 apresenta essa classificação.

A organização com base no Modo de Limitar o Otimismo justifica-se pelo fato de que alguns protocolos limitam o otimismo sem fazer análise do estado da simulação ou do processo lógico (por exemplo, com respeito ao número de *rollbacks* e eficiência), mas adaptam-se durante a execução. A seção 3.3 discute essa abordagem.

3.1 Organização Segundo as Características de Implementação

A proposta original do *Time Warp* define a noção de tempo virtual e fornece as diretrizes de como efetuar a sincronização entre os processos lógicos usando a abordagem otimista com *rollbacks* e salvamento de estados. Porém, no trabalho de Jefferson não estão especificadas, para citar alguns exemplos, a forma pela qual as antimensagens devem ser enviadas, a maneira como o *rollback* deve ser efetuado ou ainda como e quando salvar as informações relativas aos estados dos processos lógicos e como efetuar o cálculo do GVT (*Global Virtual Time*). Isso implica que os projetistas e pesquisadores contam com uma gama variada de opções, além de poderem criar suas próprias soluções no desenvolvimento de seus ambientes de simulação que fornecem suporte ao *Time Warp*. As seções a seguir descrevem cada uma das classes apresentadas na classificação da figura 1.

3.1.1 Organização com Base no Salvamento de Estados

O *Time Warp* é um protocolo de sincronização otimista que permite que a simulação distribuída execute até que ocorra um erro de causa e efeito. Quando isso ocorre, *Time Warp* executa um *rollback* para retornar a simulação para um estado consistente. A execução do *rollback*, juntamente com o retorno da simulação a um estado consistente, exige alguns procedimentos que podem afetar o desempenho da simulação.

O procedimento de salvamento de estados no *Time Warp* pode ser executado por uma das seguintes abordagens (figura 4) [12]:

Copy state saving: todo o estado do processo é salvo após a execução de cada evento;

Sparse state saving: todo o estado do processo lógico é salvo periodicamente e qualquer estado pode ser recuperado através da restauração do último estado salvo antes do evento afetado pelo *rollback* e pela re-execução de eventos intermediários. Pode ser fixo (o intervalo de tempo entre os salvamentos de estado é constante durante toda a execução da simulação) ou adaptativo (o intervalo de tempo entre os salvamentos de estado é escolhido dinamicamente). Alguns métodos descritos na literatura são o Método de Rönngren Baseado no Tempo de Execução, o Método de Rönngren Baseado no Consumo de Memória, o Método de Fleischmann-Wilsey e o Método para de Chung-Xu [4] [17] [23] [24];

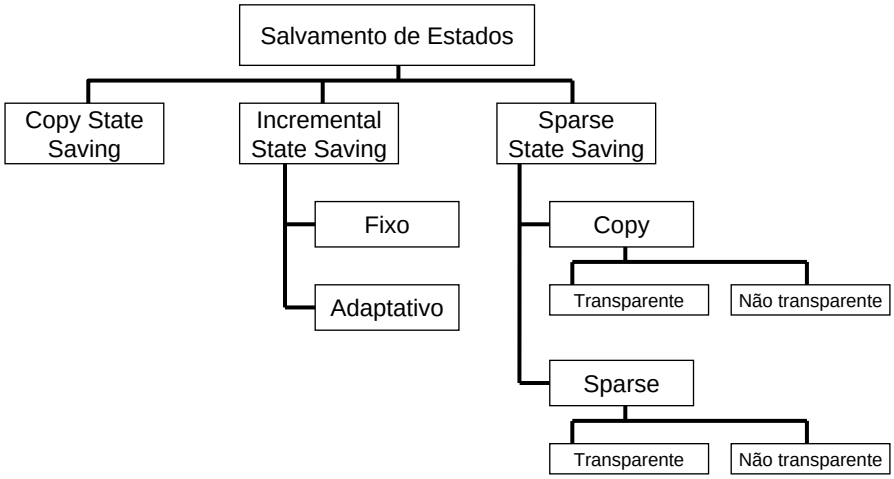


Figura 4: Organização por Salvamento de Estados

Incremental state saving: somente as partes do estado do processo lógico que foram modificadas são salvas. Pode ser dividido em transparente ou não transparente. No caso não-transparente, todas as variáveis de estado que foram modificadas devem ser identificadas pelo usuário da simulação e atualizadas. No caso transparente, as variáveis de estado que sofreram modificação são atualizadas automaticamente pelo sistema de simulação. A implementação pode optar pelo caso *Copy* e salvar as partes modificadas do estado a cada alteração, ou determinar intervalos de tempo para efetuar o salvamento, identificando-se ao caso *Sparse*. Compose (*Conservative Optimistic and Mixed Parallel-oriented Simulation Environment*) é um exemplo de utilização de *Incremental state saving* [13]

3.1.2 Organização com Base no Cálculo do GVT

O grande problema associado à tarefa de salvar estados consiste na utilização do espaço de memória disponível. Dessa forma, outro aspecto importante na implementação de um sistema *Time Warp* reside na adoção de alguma técnica para descartar os estados que foram salvos mas que não serão mais necessários. O valor do GVT garante que os estados salvos com marcas de tempo menores podem ser descartados, pois não ocorre *rollback* para estados anteriores ao GVT.

Outra razão importante para efetuar o cálculo do GVT em uma simulação distribuída consiste em oferecer suporte para a simulação distribuída interativa (DIS - *Distributed Interactive Simulation*). É permitido que apenas os eventos seguros (suas marcas de tempo são menores do que o GVT e, portanto, não irão sofrer *rollback*) podem liberar informação para o mundo externo.

O fator mais importante e que torna o cálculo do GTV uma tarefa difícil é o fato de existirem mensagens/antimensagens circulando pela rede de comunicação. Essas mensagens/antimensagens devem ser consideradas no momento de se determinar qual a menor marca de tempo da simulação distribuída.

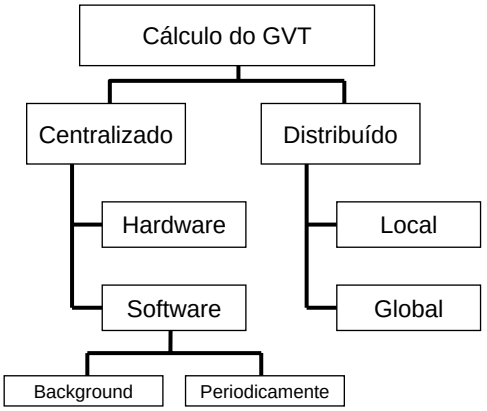


Figura 5: Organização com Base no Cálculo do *GVT*

Existem diferentes abordagens para o cálculo do valor do GVT (figura 5). Pode-se dividir essas abordagens em centralizadas ou distribuídas. Na abordagem centralizada o valor do GVT é calculado por um processo lógico (*software*) ou *hardware* específico [6]. A vantagem de se ter um *hardware* específico para efetuar tal tarefa é que existe pouca influência no desempenho da simulação, porém tem-se um aumento no custo e na complexidade.

A abordagem de um processo lógico específico (*software*) é a mais utilizada na literatura, e nesse caso tem-se o problema de minimizar o impacto do algoritmo de cálculo no desempenho da simulação distribuída. Esse algoritmo pode ser executado em *background* (concorrendo assim com os demais processos lógicos) ou pode ser iniciado periodicamente (por exemplo, quando um processo lógico necessita efetuar *fossil collection* para poder continuar a execução dos eventos). Nesse caso os processos lógicos devem esperar o novo valor do GVT [30].

Uma vantagem da abordagem distribuída para o cálculo do GVT é a ausência de sobrecarga em um único processo lógico, pois o trabalho é distribuído por todos. Essa abordagem também pode ser aplicada de forma local, onde os processos lógicos vizinhos calculam seu GVT que depois é comparado aos demais GVTs das outras vizinhanças e o menor é eleito o GVT, ou então de forma global, onde um único valor de GVT é calculado considerando-se todos os processos lógicos. *Speed GVT* [30] é um exemplo de abordagem distribuída global.

3.1.3 Organização com Base na Recuperação de Erro de Causa e Efeito

Quando ocorre um erro de causa e efeito na simulação distribuída, o protocolo de sincronização precisa corrigí-lo e reiniciar a simulação a partir de um ponto seguro no tempo. Existem duas formas de se corrigir um erro de causa e efeito (figura 6). Na proposta de Jefferson [10] os valores dos estados pelos quais o processo lógico passa são salvos e a simulação pode ser restaurada para um estado seguro através do *rollback*, que copia de volta os valores do estado salvo para as variáveis da simulação. GTW [9] e *Warped* [19] são exemplos de sistemas de simulação distribuída *Time Warp* que utilizam o *rollback* e salvamento de estados para recuperar erros de causa e efeito.

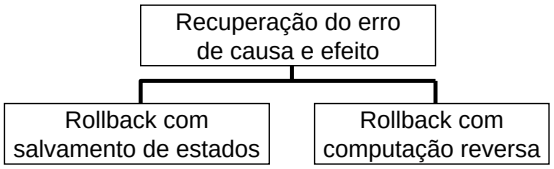


Figura 6: Organização com Base na Recuperação de Erro de Causa e Efeito

Outra forma de se recuperar um erro de causa e efeito consiste em implementar Computação Reversa, proposta por Carothers [3] [2]. Como exemplo tem-se o sistema ROSS (*Rensselaer's Optimistic Simulation System*) [1]. Nessa técnica, o *rollback* é realizado pela execução das operações inversas relativas às operações individuais que são efetuadas durante a execução de um evento. O sistema garante que as operações inversas restauram o estado da simulação para os valores que antecedem a execução do evento. A vantagem dessa técnica reside na pequena quantidade de informação de controle que precisa ser salva, em relação ao tamanho de todo o estado [2].

3.1.4 Organização com Base no Cancelamento de Mensagens

Outro fator importante que deve ser levado em consideração nas implementações baseadas no *Time Warp* relaciona-se às mensagens que são enviadas de forma errada quando ocorre um erro de causa e efeito. Essas mensagens devem ser canceladas quando um *rollback* é executado, através do envio explícito de antimensagens (Cancelamento Indireto) ou do cancelamento das mensagens enviadas diretamente nas filas — dos outros processos lógicos, em caso de memória compartilhada, ou na fila de saída do próprio processo, no caso de memória distribuída (Cancelamento Direto) (figura 7). Em arquiteturas de memória compartilhada, quando a execução de um evento resulta no envio de um novo evento para um processo lógico remoto, um apontador para este novo evento é mantido juntamente com o evento executado, na estrutura de dados do processo emissor. Isso elimina a necessidade do envio explícito de antimensagens [2]. Quando se está utilizando uma arquitetura com memória distribuída, o cancelamento ocorre no buffer de envio (NIC - *Network Interface Card*), como no *Early Cancellation* [15].

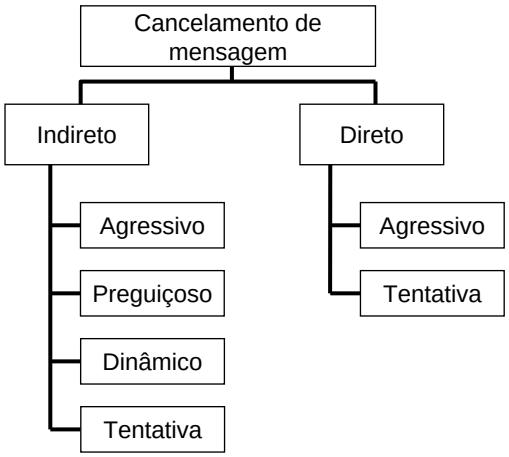


Figura 7: Organização com Base no Cancelamento de Mensagens

No Cancelamento Indireto (com uso de antimensages) as abordagens utilizadas são Cancelamento Agressivo, Cancelamento Preguiçoso, Cancelamento Dinâmico e Cancelamento por Tentativa [11]. No Cancelamento Direto (sem uso de antimensagens, pode-se utilizar as abordagens Agressivo e Tentativo.

Quando Cancelamento Agressivo é adotado e um processo lógico sofre um *rollback* para o tempo t, são enviadas antimensagens imediatamente para anular todas as mensagens enviadas com marca de tempo maior do que t. No Cancelamento Preguiçoso o processo lógico espera para ver se a reexecução dos eventos executados fora de ordem produz as mesmas mensagens. Em caso afirmativo não é necessário enviar antimensagens e no caso contrário as antimensagens são enviadas.

O Cancelamento Dinâmico permite que cada processo lógico decida qual estratégia de Cancelamento, Agressivo ou Preguiçoso, vai utilizar.

No Cancelamento em Tentativa, utilizado no *Tentative Time Warp*, um grupo de mensagens enviadas para um processo lógico pode ser cancelado por uma única antimensagem.

3.1.5 Organização com Base em Fossil Collection

As versões tradicionais do *Time Warp* implementam *fossil collection* através da comparação da marca de tempo da informação que foi salva com o valor do GVT. Tudo o que for anterior a esse tempo pode ser descartado e a memória utilizada pode ser liberada, uma vez que os *rollbacks* só irão ocorrer no máximo até o valor do relógio global. Isso implica que o cálculo do GVT deve ser efetuado regularmente (ou sempre que um processo lógico necessita espaço em memória) para que os processos lógicos consigam liberar espaço de memória e a simulação possa prosseguir. Essa abordagem pode ser considerada conservativa uma vez que os processos lógicos somente liberam a memória ocupada depois de ter certeza de que os estados não serão mais necessários para garantir a correta execução dos eventos da simulação e isso só acontece quando o valor de GVT é calculado [12]. Um processo lógico que utiliza mais espaço em memória do que os demais e precisa liberá-lo pode afetar o desempenho da simulação com sucessivos pedidos para que o cálculo do GVT seja efetuado. A figura 8 mostra a classificação dos protocolos segundo a forma de efetuar *fossil collection*.

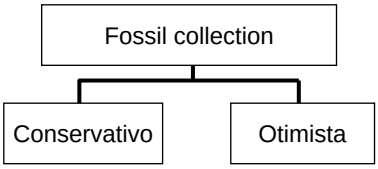


Figura 8: Organização com Base em Fossil Collection

A abordagem OFC (*Optimistic Fossil Collection*) [32] permite que os processos lógicos efetuem a tarefa de liberar espaço de memória de maneira independente, sem trocar informação com os demais processos. Cada um deles faz uma previsão do valor do GVT, com isso pode-se determinar uma estimativa do número x de estados de forma que a probabilidade de que um *rollback* atinja um número de estados maior do que x seja um fator de risco. Esse fator de risco é utilizado para determinar a agressividade da tarefa de efetuar *fossil*

collection. Dessa forma, a liberação de espaço em memória é feita de forma otimista, pois cada processo lógico considera que os estados mais antigos não serão mais necessários e, ao contrário da abordagem conservativa, não fica esperando o cálculo do GVT.

3.2 Organização Segundo as Características do Protocolo de Sincronização

Essa classificação agrupa os protocolos conforme apresentado na figura 2. O protocolo Básico consiste no *Time Warp* proposto por Jefferson [10] sem nenhuma modificação ou otimização quanto ao otimismo. Os protocolos Estendidos consistem em variantes do básico *Time Warp* que procuram reduzir o número de erros de causa e efeito e assim melhorar o desempenho da simulação distribuída.

Os protocolos Estendidos procuram aproveitar as vantagens do *Time Warp* básico como, por exemplo, explorar o paralelismo inerente à aplicação e também eliminar ou ao menos amenizar suas desvantagens (como o excesso de otimismo que pode levar a computações errôneas). A idéia básica consiste em limitar esse excesso de confiança de que a computação de um evento está sendo efetuada corretamente no tempo e assim diminuir a quantidade de erros de causa e efeito e até mesmo a necessidade por espaço livre em memória para armazenar os estados salvos. Assim como o *Time Warp* básico, todos os protocolos que estendem a funcionalidade do *Time Warp* fazem uso das Características de Implementação descritas anteriormente.

Os protocolos Estendidos podem ser divididos em três grandes grupos, que envolvem aqueles que introduzem técnicas para limitar o otimismo na sincronização entre os processos lógicos, aqueles que se preocupam em otimizar o consumo do espaço de memória disponível para que os processos lógicos efetuem salvamento de estados (limitando assim indiretamente o otimismo), e aqueles que se preocupam com o escalonamento de processos lógicos que executam em um mesmo elemento de processamento (o processo lógico com menor chance de sofrer *rollback* pode executar) (figura 9). Essas duas últimas visões dos protocolos podem ser utilizadas em conjunto com a limitação do otimismo [12].

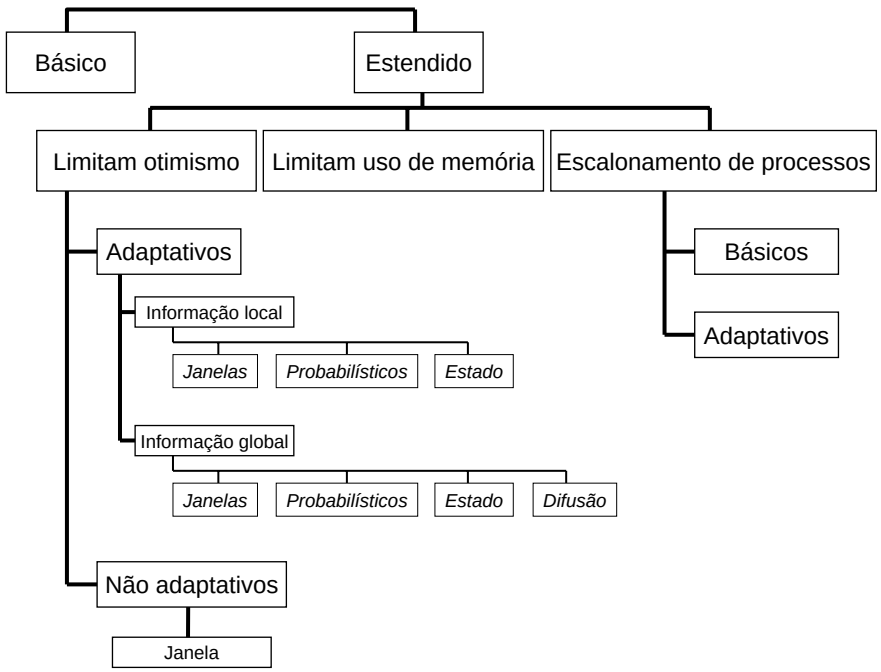


Figura 9: Organização segundo as Características do Protocolo de Sincronização

3.2.1 Protocolos que Limitam o Otimismo

Esses protocolos podem ser classificados de acordo com a maneira pela qual limitam o otimismo e pela forma com que a técnica utilizada para limitação se comporta no decorrer da simulação distribuída (isto é, se existe adaptação em relação a mudanças no comportamento da simulação ou não) resultando em protocolos Adaptativos ou Não Adaptativos [5].

A maioria dos protocolos existentes pode ser classificada como adaptativa, isto é, utilizam a situação corrente da simulação distribuída para decidir quando o otimismo deve ser controlado. Os vários protocolos Adaptativos diferem no conjunto de informações que são utilizadas para avaliar o comportamento da simulação até o momento, isto é, quais dados e como esses dados são utilizados. Dessa forma, os protocolos

podem ser divididos em Locais (a tomada de decisão leva em consideração somente as informações locais ao processo lógico) ou Globais (a decisão em um processo lógico considera informações relativas a outros processos lógicos).

Os protocolos Não Adaptativos limitam o otimismo de forma estática, isto é, não existe alteração durante a execução da simulação distribuída. Alguns protocolos baseados em janelas se encaixam nesta categoria.

Dependendo da implementação, diferentes protocolos podem ser classificados como Locais ou Globais ou mesmo como sendo Não Adaptativos, como mostrado na figura 9 e discutido a seguir.

Protocolos Baseados em Janelas de Tempo: permitem que os processos lógicos executem somente aqueles eventos cujas marcas de tempo são inferiores a um determinado valor limite (limite da janela de tempo). Exemplos de implementações não adaptativas incluem *Bounded Time Warp*, *Moving Time Windows* [7] e *Window-based Throttling* [21]. Exemplos de protocolos onde os limites das janelas se adaptam dinamicamente às condições da simulação incluem *Adaptive Time Warp* [7], *Breathing Time Warp* [29], *Adaptive Bounded Time Window* [5], UDS [20] e *Adaptive Throttle Scheme* [31]. Os protocolos baseados em janelas diferem com relação ao método utilizado para determinar o tamanho da janela e se os processos lógicos compartilham a mesma janela (informação global) ou se existe uma janela particular para cada um (informação local). Como exemplo de protocolo com janelas locais tem-se *Breathing Time Warp* e protocolos com janelas globais têm-se *Window-based Throttling* e *Bounded Time Warp*;

Protocolos Probabilísticos: esses protocolos se baseiam em análises probabilísticas para prever o comportamento futuro dos processos lógicos e assim tomar decisões sobre a sincronização dos mesmos. Como exemplo pode-se citar MIMDIX, Protocolo Probabilístico baseado em Custos, Protocolo *Probabilistic Distributed Discrete Event Simulation* e o Protocolo PADOC;

Protocolos Baseados no Estado: dependendo das condições da simulação os processos lógicos são impedidos de executar eventos e assim não avançam seus LVTs (*Local Virtual Time*). Como exemplo tem-se *Penalty based Throttling*, *Length-based Blocking* e *Composite Elsa*, nos quais as informações de estado são utilizadas localmente por cada processo lógico, e *State Query Time Warp* e os Protocolos Adaptativos NPSI [27], nos quais as decisões baseiam-se no estado global da simulação. É importante ressaltar que o protocolo *Penalty based Throttling* é definido por Das como baseado em Estado local, e por Snirivasan como baseado em Penalidade. Nesta classificação, entende-se que a penalidade sofrida por um processo lógico está associada a informações relativas ao seu estado e, portanto, justifica-se a inclusão do protocolo como baseado em estado.

Protocolos Baseados em Difusão: têm como princípio básico a difusão de informação entre os processos lógicos, com o objetivo de parar computações errôneas. Pode-se considerar que tais protocolos são de abordagem global, porque a informação difundida pode afetar todos os processos da simulação, e adaptativos, porque seu comportamento pode mudar conforme a simulação apresenta um número maior ou menor de erros de causa e efeito. Como exemplo tem-se os protocolos *Wolf*, *Filter* e *Sfilter* [16]. Os protocolos *Wolf* e *Filter*, classificados por Snirivasan como baseados em conhecimento, são aqui classificados como baseados em difusão. Das os classifica apenas com limitantes do otimismo no *Time Warp*, sem aprofundar a maneira pela qual esta limitação acontece.

3.2.2 Protocolos que Controlam o Uso do Espaço de Memória

Essa classificação inclui as técnicas de limitar o uso do espaço de armazenamento disponível, isto é, o progresso dos processos lógicos no tempo de simulação é controlado de acordo com a disponibilidade de espaço em memória (para salvar estados). Esses protocolos podem ser utilizados em conjunto com os protocolos que limitam otimismo.

Ressalta-se que Das classifica os protocolos de gerenciamento de memória como protocolos adaptativos baseados no estado global da simulação. Nesta classificação, preferiu-se agrupar os protocolos de controle da memória disponível aos processos lógicos em um grupo à parte. Todos eles são adaptativos e tomam decisões com base no estado da simulação (falta de espaço para armazenar novos estados), porém são os únicos que se preocupam com a memória finita da plataforma.

A aplicação desses protocolos é atrativa devido a vários motivos. Primeiro, porque a utilização de espaço de armazenamento é um ponto crítico nos protocolos otimistas. Segundo, porque eles efetuam um controle indireto sobre o progresso da simulação, evitando *rollbacks*. Exemplos desses mecanismos são *Adaptive Memory Management Protocol* e *Cancelback Protocol* [7].

3.2.3 Protocolos que Utilizam Escalonamento de Processos Lógicos

A principal forma de se evitar que os processos lógicos no *Time Warp* executem eventos de forma totalmente otimista é através do controle direto desse otimismo. O tratamento indireto desse problema também tem sido estudado na literatura, como é o caso do escalonamento de processos lógicos.

Em simulações distribuídas que envolvem um grande número de processos lógicos, é provável a atribuição de mais de um processo lógico para um mesmo elemento de processamento. Assim, pode-se utilizar um algoritmo de escalonamento de processos lógicos de forma que aqueles com menores chances de executarem eventos fora de ordem tenham prioridade.

Esses algoritmos de escalonamento podem ser divididos em Básicos ou Estendidos. Os básicos caracterizam-se pelo fato de escalonarem sempre o processo lógico com menor valor de LVT (*Lowest-Local-Virtual-Time-First*) ou aquele com menor valor de marca de tempo do próximo evento a ser executado (*Lowest-Timestamp-First*).

Os algoritmos estendidos procuram avaliar qual processo lógico tem menor chance de sofrer *rollback* ao executar o próximo evento. Esse processo lógico é escolhido para executar. Exemplos incluem *Adaptive Control based Scheduling*, *Service Oriented Scheduling*, *Probabilistic Scheduling*, *State based Scheduling*, *Grain Sensitive Scheduling* e *Aggressiveness/Risk Effects based Scheduling* [18].

3.3 Organização Segundo o Modo de Limitar o Otimismo

Os processos lógicos podem moldar o seu comportamento em virtude das características da simulação, como por exemplo o aumento do número de *rollbacks*. Pode-se classificar os protocolos limitantes do *Time Warp* em Preventivos e Corretivos. Os primeiros se propõem a prevenir a ocorrência de rollbacks, como por exemplo *Length-based Blocking* e os protocolos baseados em Janelas Não Adaptativos. Os protocolos Corretivos, como o próprio nome indica, corrigem o comportamento do processo lógico em função do aumento do número de erros de causa e efeito. Como exemplo têm-se *Penalty-based Throttling* e os protocolos Adaptativos.

4 CONCLUSÃO

Existem duas classificações na literatura que buscam agrupar os protocolos de sincronização (apresentadas na seção 2). Porém, essas classificações não levam em consideração as características da implementação, que podem influenciar em um estudo de desempenho entre protocolos de sincronização. Além disso, uma delas classifica protocolos que inserem otimismo em protocolos conservativos, e ambas tratam os protocolos de gerenciamento de memória juntamente com outros protocolos, apesar das características únicas que apresentam. Agrupadas e avaliadas as formas de se limitar o otimismo, a classificação apresentada em [25] [12] foi detalhada.

Os principais protocolos de sincronização descritos na literatura e que têm como objetivo principal limitar o excesso de otimismo no *Time Warp* foram divididos em três grandes grupos, que se preocupam com as Características de Implementação, Mecanismo de Sincronização e o Modo de Limitar o Otimismo. Cada um desses grupos apresenta uma hierarquia de subgrupos.

Agradecimentos

Os autores agradecem à FUNDUNESP, pelo apoio financeiro, e ao Programa de Mestrado em Ciência da Computação do Departamento de Computação e Estatística da Universidade Federal de Mato Grosso do Sul, do qual a Profa. Renata Lobato é colaboradora.

Referências

[1] C. D. Carothers, D. Bauer, and S. Pearce. ROSS: a high-performance, low memory, modular time warp system. In *Proceedings of the fourteenth workshop on Parallel and distributed simulation*, pages 53–60, Bologna, Italy, 2000. IEEE Computer Society.

[2] C. D. Carothers, K. S. Perumalla, and R. M. Fujimoto. The effect of state-saving in optimistic simulation on a cache-coherent non-uniform memory access architecture. In *Proceedings of the 1999 winter simulation conference*, pages 1624–1633, Phoenix, Arizona, United States, Dec 1999.

[3] C. D. Carothers, K. S. Perumalla, and R. M. Fujimoto. Efficient optimistic parallel simulations using reverse computation. In *Proceedings of the thirteenth workshop on Parallel and distributed simulation*, pages 126–135, Atlanta, Georgia, United States, 1999. IEEE Computer Society.

[4] M. J. Chung and J. Xu. An overhead reducing technique for Time Warp. In *Proceedings of the sixth IEEE International Workshop on Distributed Simulation and Real-Time Applications*, pages 95–102, Texas, United States, Oct 2002. IEEE Computer Society.

[5] S. R. Das. Adaptive protocols for parallel discrete event simulation. In *Proceedings of the 28th conference on Winter simulation*, pages 186–193, Coronado, California, United States, 1996. ACM Press.

[6] L. M. D'Souza, X. Fan, and P. A. Wilsey. Modifications to the pGVT algorithm to eliminate acknowledgement messages and improve the GVT broadcast frequency. In *Proceedings of the World Congress on Systems Simulation: Conference on Parallel and Distributed Simulation*, pages 288–292, Coronado, California, United States, Sep 1997.

[7] A. Ferscha. Probabilistic adaptive direct optimism control in Time Warp. In *Proceedings of the 9th workshop on Parallel and distributed simulation*, pages 120–129, Lake Placid, New York, United States, 1995. IEEE Computer Society.

[8] A. Ferscha and G. Chiola. Self-adaptive logical processes: the probabilistic distributed simulation protocol. In *Proceedings of the 27th Annual Simulation Symposium*, pages 78–88, La Jolla, California, United States, 1994. IEEE Computer Society.

[9] R. M. Fujimoto. *Parallel and Distributed Simulation Systems*. John Wiley and Sons, Inc., 2000.

[10] D. R. Jefferson. Virtual Time. *IEEE Transactions on Programming Languages and Systems*, 7(3):404–425, 1985.

[11] N. Kalantery. Tentative Time Warp. In *Proceedings of the 3rd International Euro-Par Conference*, volume 1300 of *LNCS*, pages 458–467. Springer-Verlag, 1997.

[12] R. S. Lobato, R. S. Ulson, M. J. Santana, and R. H. C. Santana. A revised taxonomy for time warp based distributed synchronization protocols. In *Proceedings of the 8th IEEE Internation Symposium on Distributed Simulation and Real Time Applications*, Budapest, Hungary, 2004. IEEE Computer Society. To appear.

[13] R. A. Meyer, J. M. Martin, and R. L. Bagrodia. Slow memory: the rising cost of optimism. In *Proceedings of the fourteenth workshop on Parallel and distributed simulation*, pages 45–52, Bologna, Italy, 2000. IEEE Computer Society.

[14] J. Misra. Distributed discrete-event simulation. *ACM Computing Surveys*, 18(1):39–65, 1986.

[15] R. Noronha and N. B. Abu-Ghazaleh. Early cancellation: an active nic optimization for time-warp. In *Proceedings of the sixteenth workshop on Parallel and distributed simulation*, pages 43–50, Washington, D.C., United States, 2002. IEEE Computer Society.

[16] A. Prakash and A. Subramanian. An efficient optimistic distributed simulation scheme based on conditional knowledge. In *Proceedings of the 6th Workshop on Parallel and Distributed Simulation*. IEEE Computer Society, 1992.

[17] F. Quaglia. Event history based sparse state saving in Time Warp. In *Proceedings of the twelfth workshop on Parallel and distributed simulation*, pages 72–79, Banff, Alberta, Canada, 1998. IEEE Computer Society.

[18] F. Quaglia and V. Cortellessa. Grain sensitive event scheduling in time warp parallel discrete event simulation. In *Proceedings of the fourteenth workshop on Parallel and distributed simulation*, pages 173–180, Bologna, Italy, 2000. IEEE Computer Society.

[19] R. Radhakrishnan, L. Moore, and P. A. Wilsey. External adjustment of runtime parameters in time warp synchronized parallel simulators. In *Proceedings of the eleventh International Parallel Processing Symposium*, pages 260–266, Geneva, Switzerland, 1997. IEEE Computer Society.

[20] H. Rajaei, R. Ayani, and L.-E. Thorelli. The local time warp approach to parallel simulation. In *Proceedings of the seventh workshop on Parallel and distributed simulation*, pages 119–126, San Diego, California, United States, 1993. ACM Press.

[21] P. L. Reiher, F. Wieland, and D. Jefferson. Limitation of optimism in the time warp operating system. In *Proceedings of the 21st conference on Winter simulation*, pages 765–770, Washington, D.C., United States, 1989. ACM Press.

[22] P. F. Reynolds, C. F. Weight, and J. R. Fidler. Comparative analyses of parallel simulation protocols. In *Proceedings of the 21st conference on Winter simulation*, pages 671–679. ACM Press, 1989.

[23] R. Rönngren, L. Barriga, and R. Ayani. An incremental benchmark suite for performance tuning of parallel discrete event simulation. In *Proceedings of the 29th Hawaii International Conference on System Sciences*, 1996.

[24] H. M. Soliman. On the selection of the state saving strategies in Time Warp parallel simulators. *TRANSACTIONS of the Society for Computer Simulation International*, 16(1):32–36, Mar 1999.

[25] R. Spolon, M. J. Santana, and R. H. C. Santana. Distributed simulation, Time Warp and its variations: Taxonomy and performance evaluation issues. In *Proceedings of the 13th European Simulation Multiconference*, pages 220–227, Warsaw, Poland, 1999. The Society for Modeling and Simulation, Europe Council.

[26] S. Srinivasan and P. F. Reynolds, Jr. Adaptive algorithms vs. Time Warp: an analytical comparison. In *Proceedings of the 27th conference on Winter simulation*, pages 666–673, Arlington, Virginia, United States, 1995. ACM Press.

[27] S. Srinivasan and P. F. Reynolds, Jr. NPSI adaptive synchronization algorithms for PDES. In *Proceedings of the 27th conference on Winter simulation*, pages 658–665, Arlington, Virginia, United States, 1995. ACM Press.

[28] J. S. Steinman. SPEEDES: Synchronous parallel environment for emulation and discrete event simulation. In *Proceedings of the SCS Multiconference on Advances on Parallel and Distributed Simulation*, pages 95–103. ACM Press, 1991.

[29] J. S. Steinman. Breathing Time Warp. In *Proceedings of the seventh workshop on Parallel and distributed simulation*, pages 109–118, San Diego, California, United States, 1993. ACM Press.

[30] J. S. Steinman, C. A. Lee, L. F. Wilson, and D. M. Nicol. Global virtual time and distributed synchronization. In *Proceedings of the 9th workshop on Parallel and distributed simulation*, pages 139–148, Lake Placid, New York, United States, 1995. IEEE Computer Society.

[31] S. C. Tay, Y. M. Teo, and S. T. Kong. Speculative parallel simulation with an adaptive throttle scheme. In *Proceedings of the eleventh workshop on Parallel and distributed simulation*, pages 116–123, Lockenhaus, Austria, 1997. IEEE Computer Society.

[32] C. H. Young, N. B. Abu-Ghazaleh, and P. A. Wilsey. OFC: A distributed fossil-collection algorithm for Time-Warp. In S. Kutten, editor, *Proceedings of the 12th International Symposium on Distributed Computing*, volume 1499 of *LNCS*, page 408, Andros, Greece, 1998. Springer-Verlag.

Artículos de CIESC/CIESC's articles

Utilização das Idéias de Piaget como Suporte para o Ensino de Sistemas Operacionais

José Augusto Fabri

Alexandre L´Erário

Departamento de Engenharia de Produção – Escola Politécnica Universidade de São Paulo.
Av. Professor de Almeida Prado, Trav. 2, nº 128 CEP 05508-900 Cidade Universitária São Paulo – Brasil Tel. 55 11 3091-5363

Fundação Educacional do Município de Assis – Centro de Pesquisas em Informática
Av. Getúlio Vargas S/N – Assis SP – Brasil
CEP 19800-000 Tel. 55 18 3302 1055

Faculdade de Tecnologia de Ourinhos
Avenida Vitalino Marcusso S/N – Ourinhos SP – Brasil
CEP 19900-000 Tel. 55 14 3326 3031

fabri@femanet.com.br alexe@femanet.com.br

Abstract. To develop good projects in the area of operating systems demands qualification of the developers, it is obligation of the university to form good professionals to assist these demands. With base in this statement this work presents Jean Piaget's ideas as support for the development of operating systems project. The main objective of this article is to show that the Piaget's ideas can contribute with the work methodology of teachers that act in the discipline of operating systems.

Resumo. Desenvolver bons projetos na área de sistemas operacionais exige qualificação dos projetistas e desenvolvedores, sendo assim, é papel dos cursos superiores de computação formar bons profissionais para atender estas exigências. Com base nesta afirmação este trabalho apresenta as idéias de Jean Piaget como suporte para o desenvolvimento de projeto de sistemas operacionais. O objetivo principal deste artigo é mostrar que as idéias de Piaget podem contribuir com a metodologia de trabalho dos professores que atuam na disciplina de Sistemas Operacionais.

Key Word: Operating Systems Teach, Jean Piaget

Palavras Chaves: Ensino de Sistemas Operacionais, Jean Piaget

1. Introdução

Transmitir bons conhecimentos de sistemas operacionais aos alunos é um dos objetivos dos cursos superiores de computação e informática. Para atingir esse objetivo os professores dos cursos de ciência da computação, sistemas de informação e tecnologia em informática devem ter um trabalho diferenciado com seus alunos. Com base nessa colocação o professor de Sistemas Operacionais (SO) da Faculdade de Tecnologia de Ourinhos (FATEC-OU) apresenta neste trabalho as idéias de Jean Piaget como suporte para o desenvolvimento de projetos experimentais sistemas operacionais. O trabalho tem como objetivo aliar a idéias desenvolvidas por Piaget à prática de ensino de sistemas operacionais. Para atingir este objetivo foi elaborado um método de desenvolvimento de projetos (de caráter universitário) de sistemas operacionais.

Esse método vem sendo aplicado na Faculdade de Tecnologia de Ourinhos desde o primeiro semestre de 2002. Esta instituição possui hoje dois cursos superiores na área de informática, Tecnologia em Processamento de Dados e Análise de Sistemas e Tecnologia da Informação. No Curso de Tecnologia em Processamento de Dados são abertas 80 vagas e no curso Análise de Sistemas e Tecnologia de Informação são abertas 120 vagas. O curso de Tecnologia em Processamento possui duas disciplinas relacionadas à área de sistemas operacionais (SO I com 70 horas e SO II com 70 horas), é importante salientar que o método é aplicado na disciplina de SO II. Já o curso de Análise de Sistemas e Tecnologia da Informação possui apenas uma disciplina da área de sistemas operacionais de 140 horas.

A motivação para elaborar o método proposto neste trabalho está alicerçada nas contribuições efetuadas por Maziero (2002), entre tais contribuições destaca-se o texto abaixo:

> "(...) exige-se do aluno uma grande capacidade de construir abstrações mentais dos mecanismos envolvidos para poder compreender os aspectos mais densos da disciplina de SO. Com isso, observa-se que muitos alunos concluem a disciplina conhecendo os principais conceitos e algoritmos, mas têm muita dificuldade em integrar todos aqueles conceitos de forma coerente. Por isso, o ensino de sistemas operacionais deve ter um componente prático significativo e muito bem elaborado, que complemente as aulas teóricas e, sobretudo, permita ao aluno ter uma compreensão integrada das diversas partes de um sistema operacional (...)".

A elaboração de um componente teórico/prático que complemente a carga de conhecimento inerente ao aluno e a capacidade de construir abstração mentais nos aspectos mais densos da disciplina motivaram o desenvolvimento deste trabalhado.

Para descrever o método para o desenvolvimento de projetos de sistemas operacionais este documento está estruturado da seguinte forma: A seção 2 apresenta de forma superficial as idéias de Jean Piaget. As Idéias de Piaget para Desenvolvimento de Projetos de Sistemas Operacionais são apresentadas como estudo de caso na seção 3. Os resultados quantitativos deste método são mostrados na seção 4. A seção 5 apresenta as conclusões desse trabalho.

2. As idéias de Jean Piaget

A teoria do conhecimento desenvolvida por Jean Piaget não tem pretensões pedagógicas. Porém, esta teoria ofereceu aos professores importantes princípios para orientar sua prática (Piaget (2003)).

Segundo Dolle (2000), Piaget mostra que o ser humano estabelece desde o nascimento uma relação de interação com o meio. É nesta relação com o mundo físico e social que existe a possibilidade de desenvolvimento cognitivo.

Para Piaget (1994) e Piaget (1999) a forma de raciocinar de um ser humano passa por estágios:

- Por volta dos 2 anos o ser humano evolui do estágio sensório motor (a ação envolve órgãos sensoriais; por exemplo, sugar o leite de uma mamadeira) para o pré-operatório (capacidade de fazer uma coisa e imaginar outras; por exemplo, brincar com algum objeto e representar situações vividas anteriormente).
- Aos 7 anos o ser humano passa para o estágio operacional-concreto (relação entre os objetos e reflexão sobre o inverso dos fenômenos) é nesta fase o ser humano percebe que 3 – 1 = 2 por que sabe que 2 + 1 = 3

- Por fim o ser humano chega ao estágio operacional-formal (por volta dos 13 anos). Nesta fase o ser humano pode pensar em coisas completamente abstratas, sem necessitar da relação direta com o concreto (o ser humano é capaz de compreender conceitos como o amor e fé).

Utilizando os estágios propostos por Piaget é possível concluir que existem duas formas de conhecimento:

- o conhecimento físico, que consiste no sujeito explorando os objetos;
- e o conhecimento lógico-matemático, que consiste no sujeito estabelecendo novas relações com os objetos.

Maziero (2002), relata que Piaget desdobra a inteligência como algo dinâmico, decorrente da elaboração de mapas cognitivos que, à medida que vão sendo construídos, vão se alojando no cérebro. A inteligência do aluno não aumenta por acréscimo e sim por reorganização. Baseado nesta visão, pode-se afirmar que o aluno aprende por si, construindo e reorganizando suas próprias hipóteses sobre a realidade que o cerca.

A partir da idéia de Piaget, Bruner (1966) e outros autores em educação elaboraram a *teoria construtivista*, na qual o aprendizado é considerado um processo ativo: o aluno seleciona e transforma o conhecimento recebido, constrói hipóteses e toma decisões, contando, para isto, com um mapa cognitivo. O mapa cognitivo (esquemas, modelos mentais) fornece o significado e o significante para as experiências e permite ao indivíduo "caminhar com suas próprias pernas".

Por fim, Bruner (1966) enumera os princípios para a organização do processo de aprendizagem:

- Princípio 1: A instrução (instrução no sentido de orientar o aluno) precisa estar preocupada com as experiências e os contextos que levam o aluno a estar pronto e apto para aprender.
- Princípio 2: A instrução precisa ser estruturada para que possa ser facilmente compreendida pelo aluno, para que o mesmo construa continuamente alicerçado pelo seu conhecimento.
- Princípio 3: A instrução precisa ser criada para facilitar a extrapolação ou preencher as brechas no conhecimento (ir além do conhecimento dado).

Uma máxima teoria "piagetiana" é que o conhecimento é construído por meio da experiência. Além da construção de conhecimento por meio da experiência, Piaget mostra que o conhecimento pode ser expandido por meio da cooperação entre os agentes promotores de conhecimentos (professores, alunos e experiências reais), os atos de coação e medo são, com certeza, inibidores de conhecimento.

Utilizando a idéia desenvolvida por Piaget e o construtivismo proposto por Bruner (1966) e sabendo que o ser humano é um pesquisador em potencial, o professor pode melhorar a sua prática de ensino (Vasconcelos (1996)).

A idéia construtivista de pode ser aplicada de várias formas, neste trabalho são apresentadas duas delas:

- Levantamento de hipótese sobre a realidade, a qual reorganiza e constrói conhecimento.
- Proporcionar conflito cognitivo para que novos conhecimentos sejam produzidos.

3. As Idéias de Piaget para Desenvolvimento de Projetos de Sistemas Operacionais

A idéia de utilizar o construtivismo no desenvolvimento de projetos de Sistemas Operacionais surgiu em 2002 na Faculdade de Tecnologia de Ourinhos. A constante preocupação com a formação dos alunos para o mercado de trabalho motivou o desenvolvimento deste trabalho na disciplina de Sistemas Operacionais.

Utilizar o construtivismo no desenvolvimento de projetos de sistemas operacionais em um curso superior é simples. Para ilustrar esta simplicidade, serão apresentadas as medidas (divididas em fases) tomadas pelo professor responsável pela disciplina.

Fase 1 – Configuração do Ambiente

A configuração do ambiente tem como objetivo alicerçar a aplicação das idéias construtivistas no desenvolvimento de projetos de sistemas operacionais.

Primeiramente, os alunos da disciplina de SO do curso de Tecnologia em Processamento de Dados (40 alunos do período vespertino e 40 alunos do período noturno), foram divididos em 20 grupos de 4 alunos. Após esta divisão, cada grupo recebeu um problema hipotético da área de sistemas operacionais (vide Tabela 1) para apresentarem a solução. Após o recebimento do problema, foi estipulado um prazo de solução para os grupos

(todos os grupos tinham o mesmo prazo). O tempo pode variar de acordo com a complexidade dos problemas. É importante salientar que os grupos não recebem os mesmos problemas.

Fase 2 – Acompanhamento da Pesquisa para Solução do Problema

Ao receber o problema, cada grupo é monitorado e orientado pelo professor responsável da disciplina. O monitoramento pode ocorrer de duas formas:

Tabela 1 – Exemplo de Problemas Hipotético da Área de Sistemas Operacionais

Questões para o Curso de SO da FATEC-OU
Dada um SO, o qual trabalha com o conceito de alocação particionada dinâmica, seria possível aliar a ele a técnica de overlay? Justifique suas colocações desenvolvendo um emulador.
Qual a melhor forma de gerenciamento de memória para um sistema operacional que utiliza o escalonamento preemptivo? Desenvolva um emulador com o método escolhido.

- Monitoramento e orientação remota (não presencial): Cada grupo de alunos envia (por e-mail) ao professor artigos e títulos de livros relacionados ao problema e as possíveis soluções. O professor deve orientar se o grupo está ou não no caminho correto para solução do problema. É importante salientar que o professor estipulou um número de e-mails por grupo, cada grupo deveria enviar no 1 e-mail/semana com artigos e títulos de livros relacionados ao problema e as soluções encontradas.
- Monitoramento e orientação presencial: O professor encontra-se com os grupos no mínimo uma vez por semana, na aula ou em outros horários. Nesses encontros são esclarecidas as dúvidas e os alunos são orientados em relação à direção que devem seguir para solucionar o problema. O tempo de encontro pode variar para cada grupo. O professor também pode estipular um tempo máximo para conversar com o grupo. Na FATEC-OU o professor utilizou o horário de aula para atender os grupos.

Fase 3 – Definição de uma Solução para o Problema e Emissão de Parecer

Esgotado o tempo para solucionar o problema, os grupos devem entregar as suas soluções. Neste momento inicia-se uma segunda etapa do trabalho, onde cada grupo recebe uma solução para emitir o parecer (favorável ou desfavorável). Por exemplo, o grupo A emite o parecer para solução apresentada pelo grupo F. O parecer de cada grupo deve vir acompanhado de uma justificativa. Para emitir o parecer e a justificativa cada grupo tem um tempo, o qual deve ser estipulado pelo professor. O parecer de cada grupo também deve ser acompanhado pelo professor, para este acompanhamento o professor pode utilizar os procedimentos definidos na fase 2.

As três fases apresentadas são absorvidas pela teoria de Jean Piaget apresentada na seção 2. A Figura 1 ilustra esta absorção.

Por meio da Figura 1 é possível verificar a presença das duas idéias de Piaget para o construtivismo apresentadas na seção 2: levantamento de hipótese sobre a realidade e proporcionar conflito cognitivo para que novos conhecimentos sejam construídos. As fases são absorvidas pelas idéias de Piaget sendo que a idéia de conflito cognitivo fica sob responsabilidade da fase 3.

A idéia de conflito cognitivo é aperfeiçoada em uma outra fase, a de apresentação dos trabalhos (fase 4). Nesta fase todos os grupos apresentam os seus trabalhos. Supondo que o grupo A apresente o seu trabalho, os demais grupos ficam incumbidos de questioná-lo. O grupo A ganha um ponto para cada resposta apresentada, isso provê uma concorrência sadia entre os grupos, além de provocar o conflito cognitivo quando algum grupo não concorda com a posição do grupo A. Para motivar a elaboração de questões, o professor da disciplina de SO adotou a seguinte sistemática: Os grupos ganham um ponto para cada pergunta elaborada.

No final da fase de apresentação, os grupos que desenvolveram os melhores trabalhos são convidados a apresentar as suas soluções em uma amostra de trabalhos acadêmicos, na FATEC-OU.

É importante salientar que os alunos utilizam os seus conhecimentos para solucionar os problemas caracterizando assim a idéia de construtivismo.

Com a utilização das idéias apresentadas é possível cobrir todo conteúdo da área de sistemas operacionais: Gerenciamento de Memória, Gerenciamento de Processador, Gerenciamento de Disco, Sistemas Operacionais Multiprocessados e Distribuídos, etc. Os problemas hipotéticos apresentados aos alunos são de caráter teórico (necessidade de um estudo teórico para solução) e caráter teórico-prático (necessidade de um estudo teórico para solução, a qual dever vir acompanhada algum tipo de implementação). A obra de Tanenbaum (2003) é utilizada como livro texto para cobrir o conteúdo apresentado.

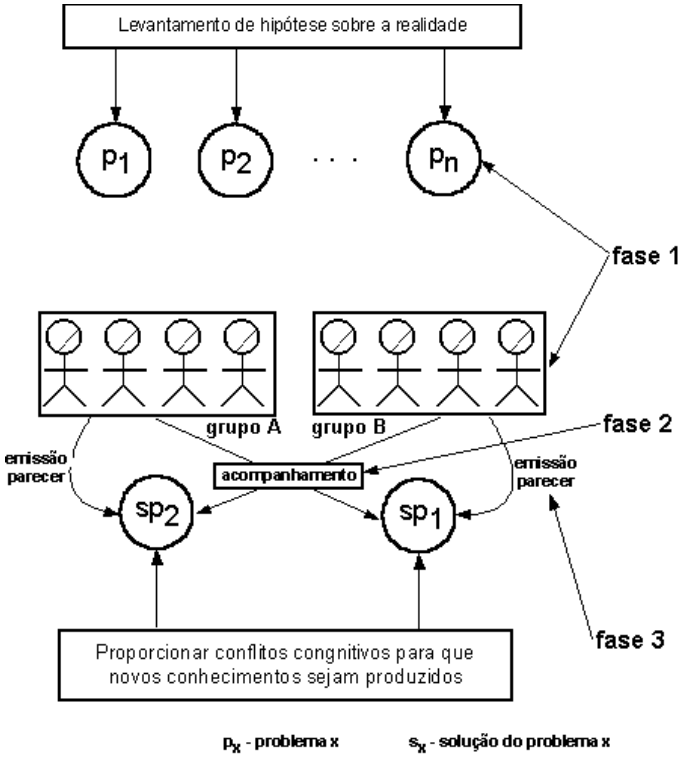


Figura 1 - Absorção das Idéias de Piaget para o Desenvolvimento de Projetos de Sistemas Operacionais.

4. Resultados Quantitativos Obtidos com a Aplicação das Idéias de Piaget.

Conforme citado no início da seção 3, a utilização das idéias construtivistas de Piaget vem sendo aplicada na FATEC-OU desde 2002. Cerca de 320 alunos passaram por esta experiência, e os resultados obtidos podem ser vistos por meio da Tabela 2.

Tabela 2 – Resultados Quantitativos Obtidos com a Aplicação da Idéias de Piaget

Atividade	Semestre/Ano				
	2/2001	1/2002	2/2002	1/2003	2/2003
Índice de Aprovação	60%	64%	70%	75%	81%
Trabalhos de Teóricos	0	2	2	4	3
Trabalhos Teóricos/Práticos	0	2	3	3	4
Apresentação de trabalhos dos alunos em Congressos de Iniciação Científica (área de SO)[1]	0	2	2	3	3
Participação dos Alunos em Programa de Iniciação Científica (área de SO)	0	4	5	7	7
[1]Congresso Nacional de Iniciação Científica, Workshop de Informática da FEMA e Workshop de informática da FATEC					

Analisando os dados apresentados na Tabela 2, verifica-se que houve um crescimento no índice de aprovação no decorrer dos semestres. No final do segundo semestre de 2003, 81% dos alunos foram aprovados, 19% reprovados sendo que 10% (dos 19%) abandonaram o curso de sistemas operacionais. Conclui-se que 9% dos alunos não absorveram o conteúdo da disciplina de sistemas operacionais. O número de apresentações de trabalhos em congressos de iniciação científica também cresceu, o que provê um diferencial na formação dos alunos. O interesse dos alunos em participar de programas de iniciação científica na área de SO também pode ser salientado na pesquisa, pois até 2001 poucos alunos queriam trabalhar com sistemas operacionais, em 2003 cerca de 7 alunos (quase 10% dos alunos matriculados no semestre) estão desenvolvendo pesquisa em SO.

Para efeito de comparação, um dos autores do artigo é responsável pela disciplina de SO na Fundacão Educacional do Município de Assis (FEMA). A FEMA possui a disciplina de SO no curso de Bacharelado em Ciência da Computação. Tal disciplina conta com uma carga em torno de 140 horas/ano. Cerca de 25 alunos estão matriculados nesta disciplina. E a mesma é oferecida no quarto ano do curso. Nesta instituição os alunos não estão imersos no método demonstrado nesse trabalho. Em virtude desse fator é possível verificar que:

- Em 2002 cerca de 27% dos alunos ficaram retidos na disciplina de SO na FEMA;
- Em 2003 cerca de 25%.

Comparando o índice de 2003 é possível verificar que a Faculdade de Tecnologia de Ourinhos, instituição onde o método é aplicado, o índice de reprovação é menor.

Na FEMA, o número de trabalhos de iniciação científica desenvolvidos na área de SO não chegou a 2% em 2003, já na FATEC-OU este índice encontra-se em torno de 10%.

Em 2003 a Faculdade de Tecnologia de Ourinhos obteve 3 trabalhos, da área de SO, publicados em congressos de iniciação científica, já a FEMA não obteve nenhum trabalho publicado relacionada a área de SO.

Esses números mostram que o método proposto neste documento pode trazer um diferencial no trabalho efetuado pelos professores da disciplina de SO.

É importante salientar que ambos professores participaram da elaboração do método, porém o mesmo necessitou ser validado, sendo assim o método foi aplicado inicialmente na Faculdade de Tecnologia de Ourinhos e posteriormente na Fundação Educacional do Município de Assis. Na FEMA o método foi colocado em prática no início de 2004.

Cabe ressaltar que não é objetivo dos autores qualificar os cursos analisados, sendo assim é possível afirmar que a formação do aluno da FEMA, em hipótese alguma, é melhor ou pior em relação a formação do aluno da FATEC-OU.

Os autores deste trabalho esperam que quando o método for aplicado na FEMA, o índice de reprovação diminua e o número de alunos publicando trabalhos em congressos de iniciação científica cresça.

O professor da disciplina dispôs de um tempo razoável para implantar este processo na Faculdade de Tecnologia de Ourinhos. No primeiro semestre de 2002 cerca de 10 horas semanais, além das aulas eram utilizadas pelo professor. Atualmente, o professor dispõe de cerca de 5 horas semanais, além das aulas (gastando cerca de 1 hora e trinta minutos respondendo e-mails). Um dos motivos desta redução foi resultado do desenvolvimento de um banco de projetos de SO.

4. Conclusões e Trabalhos Futuros.

Esse trabalho apresentou idéias de Piaget como suporte para o desenvolvimento de projetos de sistemas operacionais.

A aplicação deste método mostra que existe uma maior motivação por parte dos alunos, em relação aos conceitos da disciplina, na construção de seus conhecimentos. Como prova disso percebe-se um aumento no índice de aprovação e do interesse pelos temas relacionados à área de sistemas operacionais.

O construtivismo mostra que o aluno pode ser capaz de "caminhar com suas próprias pernas", desde que este seja bem orientado. Além de "caminhar com suas próprias pernas" o conhecimento pode tornar-se inerente ao aluno, pois a construção do conhecimento foi desenvolvida por ele mesmo.

É importante salientar que existem outros autores que tratam a questão do ensino e aprendizado, entre eles pode-se destacar Ausubel, onde este trata o aluno como um processador de informação, pois o mesmo já possui conhecimento anterior.

A opção do autor pela abordagem construtivista justifica-se por ser esta uma das abordagens mais utilizadas no ensino das ciências, notadamente no campo da engenharia [Denning (1989), Duffy, Sarevy (1995), Ben-Ari (1998)].

Um ponto fraco deste trabalho é a comparação entre os cursos de Tecnologia em Processamento de Dados e Ciência da Computação. Seria interessante aplicar a metodologia em cursos do mesmo segmento e verificar, novamente, os resultados.

Como trabalhos futuros, seria interessante aplicar o construtivismo em outras disciplinas e verificar o comportamento dos alunos em campos diferentes do conhecimento. Uma análise comparativa entre Ausubel e Piaget também se configura como meta futura deste trabalho.

Referências Bibliográficas

[1] Dolle, Jean-Marie. "Para Compreender Piaget". Ed. Agir, 1ª edição, 2000.

[2] Piaget, Jean. "O Juízo Moral das Crianças". Ed. Summus, 1ª edição. 1994.

[3] Piaget, Jean. "A Linguagem e o Pensamento da Criança". Ed. Martins Fontes Pedagogia. 1999.

[4] Piaget, Jean; D'amorim, Maria Alice Magalhaes. "Seis Estudos de Psicologia". Editora Forense Universitária. 2003

[5] Vasconcelos, Mário Sérgio. "A Difusão das Idéias de Piaget no Brasil". Ed. Casa do Psicólogo. 1ª edição 1996.

[6] Tanenbaum. Andrew S. "Sistemas Operacionais Modernos". 2ª Edição. Prentice Hall. 2003.

[7] Denning, P. (1989). "Computing as a discipline". Communications of the ACM, 32(1):9–23.

[8] Duffy, T. M. and Savery, J. R. (1995). "Problem based learning: An instructional model and its constructivist framework". Educational Technology, 35(5):31–38.

[9] Ben-Ari, M. (1998). "Constructivism in computer science education". ACM SIGCSE Bulletin, 30(2):257–261.

[10] Bruner, J. (1966). "Toward a Theory of Instruction". Havard University Press.

[11] Maziero, C. (2002) "Reflexões sobre o Ensino Prático de Sistemas Operacionais". Anais do XII Congresso Brasileiro de Computação, Workshop de Estudos em Computação. Florianópolis, julho de 2002.

Objetos de Aprendizagem na Web como Ferramentas Auxiliares para o Ensino

Juliano Schimiguel
Universidade Estadual de Campinas, Instituto de Computação
Campinas, SP, Brazil, 13084-971
juliano.schimiguel@ic.unicamp.br

Ismar Frango Silveira, Carlos Fernando Araújo Jr., Luiz Henrique do Amaral, Ivan C.A. Oliveira, Manuél Ledón, Alcides T. Barboza Jr.
Universidade Cruzeiro do Sul, Departamento de Informática
São Paulo, SP, Brazil, 08060-070
ismar.silveira@unicsul.br, carlos.araujo@unicsul.br, luiz.amaral@unicsul.br, icalcan@dglnet.com.br, mfpledon@yahoo.com, alcidestbj@yahoo.com.br

Resumo

Desde o surgimento da educação a distância, em seus primórdios, onde era e ainda é possível realizar cursos por correspondência; muito ainda tem sido discutido e desenvolvido com o intuito de aperfeiçoar as técnicas de ensino a distância. Com o avanço da área de Tecnologia de Informação (TI), essa forma de ensino-aprendizagem pôde usufruir dos recursos da Internet e da WWW (World Wide Web), tornando possível a disponibilização rápida e eficiente de conteúdos digitais na web. A área de ensino-aprendizagem auxiliada pela web é multidisciplinar e envolve disciplinas como linguagens, comunicação, letras, pedagogia, entre outras. O objetivo deste trabalho é ilustrar a aplicação de ferramentas Web na atividade de ensino-aprendizagem, considerando-se conceitos relacionados à qualidade e padronização de conteúdos digitais baseados na Web, bem como, fatores pedagógicos corelatos.

Palavras-chave: Ensino-aprendizagem baseado na web, padronização, conteúdos digitais.

1. INTRODUÇÃO

O Ensino-Aprendizagem mediado por computador ainda é um novo paradigma que sofre do mesmo tipo de discriminação ao qual estão sujeitas todas as coisas ou situações que as pessoas ainda não conhecem bem.

A educação totalmente a distância ainda está caminhando para ser aceita. Neste artigo, será tratado o ensino via Internet como uma atividade extracurricular, ou seja, como um complemento do ensino presencial.

Muitos *websites* que disponibilizam conteúdo educativo não aproveitam o potencial do computador como ferramenta didática. Esses *websites* tratam o computador simplesmente como livros eletrônicos, apenas colocando textos estáticos e questionários para o aluno imprimir e responder na forma convencional.

Para projetar e disponibilizar um ambiente de ensino a distância, são necessárias técnicas e métodos adequados, e não simplesmente a colocação do material no *website* da forma em que ele foi retirado dos livros. É sob esta ótica que será desenvolvido o presente artigo.

Este trabalho está organizado da seguinte forma: na seção 2, descrevemos os objetivos do artigo; em 3, falamos sobre o aparecimento da Educação a Distância; em 4, destacamos os esforços a serem feitos para produzir material *online*. Na seção 5, descrevemos o processo de avaliação de aprendizagem; em 6, citamos o caso do projeto WEL, entre outros casos; e, em 7, concluímos nosso trabalho.

2. OBJETIVOS

Esse artigo visa descrever a aplicação de ferramentas web na atividade de ensino-aprendizagem para auxiliar o desenvolvimento de projetos educacionais.

Devido à quantidade de alunos na sala, alguns alunos acabam não tirando todas as dúvidas quanto à matéria. Por isso, este artigo tem o objetivo de demonstrar e fornecer mais uma alternativa para estudos, com um horário mais flexível, com a correção imediata dos exercícios. Também visa oferecer passatempos como fundo educativo, além de levar o colégio para um ambiente virtual interativo.

Com o desenvolvimento de tais ferramentas, os alunos poderão tirar dúvidas de sua própria casa, assim como fazer exercícios e ter resultados instantâneos.

A disponibilização de conteúdos via Internet vem sendo questionada devido à dúvida de que um aluno a distância não tenha as mesmas oportunidades de aprendizado do que um aluno de curso presencial. Porém, é de grande vantagem o uso de ferramentas de ensino a distância como complemento do ensino presencial. A busca por meios que despertem o interesse do aluno pelo aprendizado sempre se concentrou numa parte que fugisse da teoria, onde o aluno pudesse interagir, visualizar, descobrir no dia-a-dia o que ele aprende. Com a Internet, figuras, animações, jogos e formas de interação que façam que o aluno participe do sistema, podem levá-lo a estudar mais e com prazer. Normalmente, nesse método de ensino, o horário de estudo fica a critério do aluno; o que pode causar desleixo, esquecimento, falta de interesse. Caso uma parte do material fique obscura ou complicada, o aluno tende a abandonar ou desinteressar sobre ele. No caso dessas ferramentas serem usadas como complemento, o professor pode cobrar dos alunos que eles acessem, pode disponibilizar material exclusivamente no ambiente Web; pode utilizar ferramentas que controlem o momento de acessos do aluno, etc. Muitos de nós sabemos que alguns alunos se sentem intimidados a fazerem certas perguntas na sala de aula. Com essas ferramentas, o aluno pode se sentir menos pressionado para expor suas idéias.

3. A EDUCAÇÃO A DISTÂNCIA DESDE O INÍCIO

A educação a distância surgiu com os cursos por correspondência, onde, depois de matriculado, o aluno recebia o material em casa e enviava de volta os exercícios para correção. Um exemplo são os cursos do Instituto Universal Brasileiro, desde a década de 80 até os tempos atuais.

Com a invenção da televisão e do rádio, foi repensada a forma de transmitir informação para o aprendizado. A televisão é um meio de comunicação de massa moderno e eficiente, porém os programas de televisão induzem à passividade, não à participação. Já o computador, como meio de comunicação, tende a ser interativo, o que exige uma disposição prévia do usuário em interagir com o meio.

Aproveitando-se essa interatividade, o aluno não precisa mais atuar como um simples receptor de informação, a educação a distância abre as possibilidades dos alunos interagirem com os professores e com os aplicativos.

Alves et al. (2003) afirmaram que 70% a 90% das mensagens que circulam em um ambiente virtual são contribuições dos alunos participantes, contrariamente ao que acontece em cursos presenciais onde a participação do professor é muito maior nas discussões, cabendo um percentual de apenas 20% ou menos aos alunos.

Nos últimos anos, o constante crescimento do número de cursos que se utilizam das tecnologias baseadas na web, vêm provocando uma verdadeira revolução nas mais diversas áreas de educação. A crescente popularização de tecnologias hipermídia vem provocando uma demanda cada vez maior por cursos a distância, cuja qualidade é medida

não somente pelo seu conteúdo, mas também por seu design, navegabilidade, organização e clareza das informações.

Além disso, a maior parcela dos cursos a distância produzidos e disponibilizados para acesso via Internet é composta de cursos de extensão, atualização ou treinamento, sendo muito pequeno o número de instituições educacionais voltadas à "educação formal"- escolas, faculdades e Universidades - a disponibilizarem material de apoio às disciplinas curriculares de seus cursos regulares, e quando o fazem, nem sempre seguem os critérios descritos anteriormente como indicadores da "qualidade" do material.

Um fator relevante é que uma vasta gama de recursos computacionais que podem ser utilizados na preparação de materiais em geral não são utilizados. Exemplos disso são: simulações em tempo real, modelos tridimensionais, avaliações eletrônicas, animações e gráficos, recursos de grande importância no processo de ensino-aprendizagem e que dificilmente são reproduzidos em sala de aula no ensino tradicional.

A expansão do uso das novas tecnologias da informação e comunicação (TICs) traz junto consigo um conjunto de novos paradigmas a serem analisados, ao mesmo tempo em que força a reflexão sobre antigas e bem estabelecidas estratégias que podem ou não se adequarem a esses novos paradigmas.

4. ESFORÇOS PARA PROJETAR MATERIAL ONLINE

Os estudos realizados em pesquisas indicam que o professor participa e interage pouco com um ambiente de aprendizagem. Dessa forma, dá-se a impressão de que o ensino pela Internet diminui o papel e o trabalho dele. Pelo contrário, os esforços do professor precisam ser maiores, pois ele precisa adaptar o conteúdo, os exemplos e os exercícios para a forma eletrônica. O que por um lado facilita a maneira do aluno entender e visualizar o problema, exige mais de quem elabora esse material. Assim, o professor também deve se preocupar em ter um conteúdo e uma interface agradável, para não sobrecarregar o visual e "cansar" o aluno logo na primeira página aberta; deve prender o interesse do aluno, induzí-lo a participar do ambiente e não apenas ler o que está escrito; deve levá-lo a pensar.

Por exemplo, o professor deve elaborar comentários para os casos em que o aluno escolhe as respostas erradas, incentivando-os a retornar no módulo de conteúdo; as questões devem levar o aluno a refletir antes de responder; a pergunta deve estar contextualizada ou problematizada, etc.

O texto eletrônico precisa ser dinâmico. Para tanto, é necessário reconhecer que atividades de ensino-aprendizagem e comentários de auto-avaliação são essenciais. Portanto, é muito importante desenhar atividades de forma clara, atraente e, sobretudo úteis para os alunos.

Para cada módulo de conteúdo, é importante que o professor sugíra alguns tipos de atividades que farão parte do texto. Podem ser subjetivas, de múltipla escolha, de lacunas ou outras.

5. PROCESSO DE AVALIAÇÃO DE APRENDIZAGEM

Em relação ao processo de avaliação, pouca coisa se alterou em relação a modelos pedagógicos clássicos de uso da avaliação, que a encaram como um acontecimento pontual, dentro de um dado contexto contínuo de aprendizagem, onde as respostas dadas em um certo momento são decisivas na progressão ou não do aprendiz no processo [17, 18].

Mesmo com o uso das TICs como apoio ao processo de ensino-aprendizagem a distância, é bastante comum se presenciar a implementação de modelos avaliativos antigos utilizando novas tecnologias, incorrendo no erro de propor, em um contexto de educação a distância, formas de avaliação talvez mais adequadas para situações presenciais.

No que se refere aos sistemas de suporte ao ensino-aprendizagem a distância, o processo de avaliação nem sempre é contemplado satisfatoriamente pelas ferramentas hoje disponíveis, quer seja pela limitação no que tange aos tipos possíveis de questões, ou no que diz respeito à complexidade em elaborá-las, ou mesmo à inexistência de algoritmos eficientes de processamento de linguagem natural para a análise de respostas discursivas.

É ponto pacífico a necessidade de um processo de avaliação em um contexto de ensino-aprendizagem, seja este presencial ou a distância. Deve-se discutir, todavia, uma série de pontos essenciais à definição do tipo de avaliação que se pretende estabelecer.

Uma decisão é crucial no processo de introdução do mecanismo de *e-valuation* em um contexto de ensino-aprendizagem: deve-se estabelecer qual o objetivo primordial da avaliação dentro desse contexto, ou seja, se a mesma será utilizada como mero instrumento de medida ou como referência para contribuir em futuras aprendizagens. A partir dessa análise, escolhas adequadas podem ser feitas de acordo com o critério de avaliação estabelecido.

Assim, deve-se fazer uma opção pela avaliação somativa ou formativa, respectivamente. A opção pela primeira traz para um âmbito de educação a distância, uma estratégia que por sua vez é caracterizada pela necessidade de aplicação presencial, já que se tratam de eventos pontuais com o propósito de estabelecer uma medida para o conhecimento acumulado. Já a segunda opção parte do pressuposto que através da análise de resultados de um certo aprendiz, podem

ser traçadas novas estratégias de ensino, que o conduzam a novos caminhos de aprendizagem, de maneira individualizada, servindo também como instrumento para a aprimoração do próprio curso, de forma que favoreça a reprodução de operações bem sucedidas e a reflexão sobre as que não o foram. Conclui-se então, que trata-se de um processo em andamento a ser considerado em todos os estágios da instrução, que permite o aprimoraento do curso, facilitando a adaptação dos objetos de aprendizagem às necessidades individuais e identificando falhas no planejamento e necessidade de ajustes.

O papel da avaliação formativa é o de adaptar o dispositivo pedagógico à realidade das aprendizagens dos alunos, estando presente de forma ubíqua em todos os pontos do processo de ensino-aprendizagem, pois torna-se parte do processo, e não mecanismo de medida de qualidade deste. Além disso, também serve como uma medida de qualidade em se tratando de avaliar os objetos - e não somente os agentes - de aprendizagem, de maneira singular, identificando, através do desempenho dos alunos, quais objetos são ou não eficazes na construção do conhecimento, indicando possíveis substituições, melhorias ou mudanças de estratégia pedagógica.

Ao contrário da avaliação formativa, a somativa tem como característica básica o foco no impacto pedagógico dos objetos nos agentes, geralmente através de medidas numéricas tomadas ao final de um ciclo de aprendizagem ou ao se completar um subconjunto do currículo em estudo. Assim, da mesma forma que se pode utilizá-la como uma maneira de medir o desempenho dos aprendizes, também pode servir como uma medida dos resultados obtidos - e não do processo como um todo.

Em termos de implementação das supracitadas estratégias de avaliação em um processo de ensino-aprendizagem a distância, o que ocorre é que, apesar de serem de mais rápido e fácil desenvolvimento, estratégias de avaliação somativa sofrem o problema estrutural de dependerem de um *momentum* de avaliação presencial, por se tratar de um ou mais eventos pontuais em um determinado espaço de tempo que servirão como medida da eficácia de todo um processo de ensino-aprendizagem. Logo, sua aplicação a distância incorre nas inúmeras possibilidades abertas por métodos não-lícitos que possam vir a ser aplicados para tal.

6. PROJETO WEL

Esta seção apresenta um conjunto de ferramentas de auxílio ao processo de ensino-aprendizagem, geradas a partir do Projeto WEL (*Web Engineering for Learning*) [3, 15, 16, 17, 18], que propõe o desenvolvimento de software baseado na Web, com uso das metodologias, técnicas, ferramentas e métricas de qualidade da área de *Web Engineering*, para disponibilizar material "instrucional" para o ensino-aprendizado, modelado como objetos de aprendizagem.

A definição utilizada para objetos de aprendizagem é a que os trata como qualquer entidade digital que pode ser usada, reusada ou referenciada durante um processo de aprendizagem suportado pela tecnologia [3, 16, 21]. O conceito de objetos de aprendizagem é um conceito fundamental para o desenvolvimento de conteúdos e material didático digital para ser usado em experiências e projetos de larga escala, envolvendo grande número de estudantes e disciplinas na educação formal convencional. A garantia da reusabilidade permite que um conteúdo -ou parte dele- seja usado em diferentes contextos o mesmo ocorrendo com sistemas de avaliação, simulações ou sistemas para desenvolvimento de simulações.

Pesquisas na área de Objetos de Aprendizagem [16, 21] têm contribuído na busca de padrões para o desenvolvimento de material instrucional e conteúdos digitais que sejam adaptáveis (reusáveis), genéricos e escaláveis, além de ambientes de aprendizagem virtuais que suportem tais objetos de aprendizagem com suas propriedades e características. Dentro deste contexto, tem-se buscado padrões gerais e internacionais para o desenvolvimento de objetos de aprendizagem e ambientes de aprendizagem para garantir interoperabilidade.

Desta forma, o WEL Framework consiste em um conjunto genérico e abrangente de técnicas, metodologias e ferramentas para problemas comumente vivenciados em sistemas de aprendizado a distância. Um dos primeiros testes da arquitetura realizou-se através da modelagem do tópico relativo a árvores da disciplina de Estruturas de Dados para o curso regular de Bacharelado em Ciência da Computação. A Figura 1, baseada em [18], exibe um *screenshot* de um recurso de aprendizagem criado a partir de objetos de aprendizagem recuperados a partir desses mecanismos.

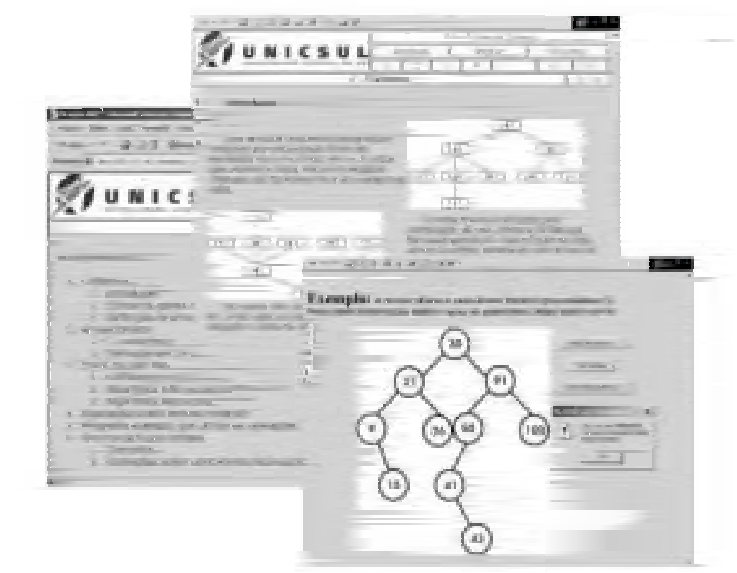


Figura 1 – Exemplos de conteúdos gerados dentro do WEL Framework

A arquitetura do WEL Framework prevê a organização dos connteúdos didáticos (ou recursos de aprendizagem) como agregados de objetos de aprendizagem com alto potencial de reuso [16]. A educação baseada em recursos (EBR) e a educação a distância (EaD) têm características singulares que podem ser otimizadas para possibilitar uma melhor experiência de aprendizagem. A EBR em ambiente Web pode ser definida como um conjunto de estratégias integradas para promover um aprendizado centrado no estudante em um contexto de uma educação em larga escala, através de uma combinação de recursos especialmente projetados para o aprendizado com mídia e tecnologias interativas. A EaD, em suas diversas formas (modelos híbridos ou completamente a distância), onde estudantes e professores podem estar separados no tempo e espaço, o papel da tecnologia e metodologia utilizados são fatores determinantes no desenvolvimento de material didático instrucional [1].

Sabemos que os processos avaliativos devem ser encarados como ferramentas de acompanhamento da construção do conhecimento por parte do aprendiz e da eficácia dos objetos de aprendizagem nessa construção, desempenhando assim um papel extremamente importante no processo de ensino-aprendizagem como um todo. Assim, um dos pontos fundamentais dentro do Projeto WEL é a elaboração de *e-valuations*, que são exercícios em caráter de avaliação formativa. Cada exercício, por sua vez, é um objeto de aprendizagem pertencente a um framework genérico e reutilizável definido em [3, 15, 16, 17, 18], sendo utilizado em todo o Projeto WEL. A arquitetura implementada é cliente-servidor, utilizando ferramentas de autoria na elaboração de interfaces para os exercícios.

Na construção do WEL Framework, ferramentas de auxílio à autoria de *e-valuations* foram utilizados na elaboração de avaliações eletrônicas, como pode ser visto na Figura 2 a seguir, que exibe um exemplo de *e-valuation* elaborado para o currículo introdutório de árvores do curso regular de Estruturas de Dados.

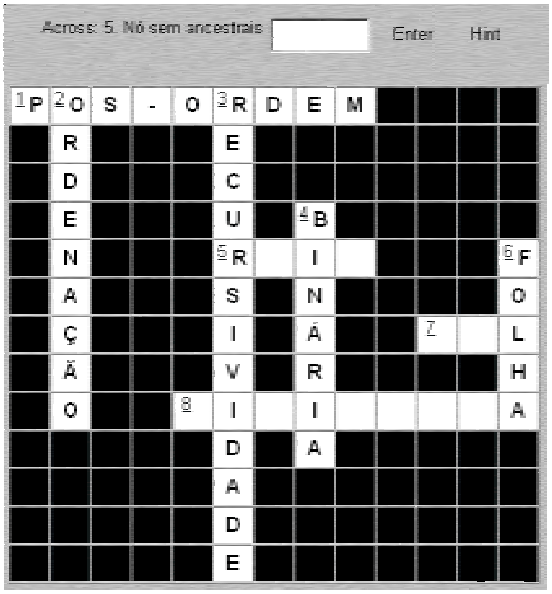


Figura 2 – Exemplo de *e-valuation* gerada pelo WEL Framework

No atual estágio do Projeto WEL, está sendo realizada a integração das ferramentas desenvolvidas com o ambiente de ensino-aprendizagem *e-Class* [5], desenvolvido pelo Núcleo de Ensino a Distância – NEAD/UNICSUL, cujo *screenshot* pode ser visto na Figura 3.

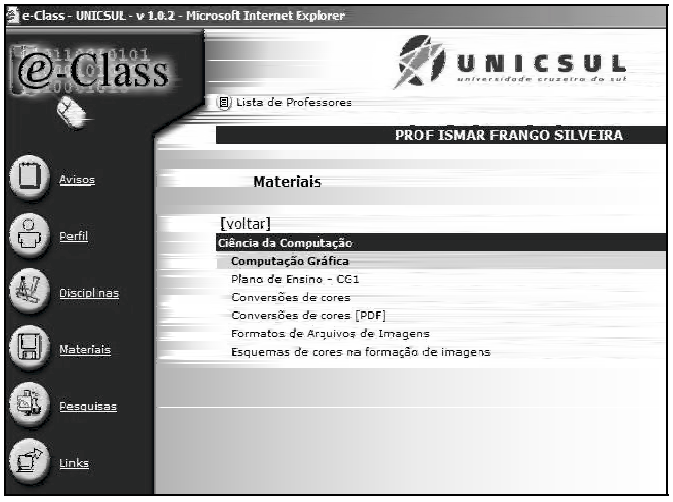


Figura 3 – *Screenshot* do e-Class

Dentro dos paradigmas considerados pelo Projeto WEL, vêm sendo desenvolvidos outros materiais de apoio ao processo de ensino-aprendizagem para uso fora do âmbito universitário, não necessariamente ligados ao e-Class ou a outro ambiente. Propôs-se implementar no site (já existente) do Colégio Floresta (São Paulo, SP) uma área de complementação do ensino em sala, onde alunos e professores possam interagir em um mesmo ambiente. Essa área, denominada "Lição de Casa", era necessário conter material que não é encontrado em livros, ou são encontrados não juntamente no mesmo lugar, nem ilustrado por figuras estáticas, como correção imediata de exercícios e ilustrações animadas simulando situações, com a preocupação de que o aluno aprenda por seu próprio interesse. Um *screenshot* demonstrando parte do trabalho já realizado pode ser visto na Figura 4 a seguir.

Figura 4 – Área "Lição de Casa", implementada no site do Colégio Floresta

Como o público-alvo diferia em alguns aspectos do público padrão do WEL, foram pesquisados alguns sites de colégios de ensino fundamental e médio que já estão caminhando para um conteúdo on-line dinâmico e interativo, com simulações e jogos. Dentre eles, destacaram-se três: o do Colégio Nobel [12], o do Colégio Santo Américo [2], e os sites do Colégio Objetivo: o primeiro voltado aos alunos do ensino médio [1] e o segundo somente com exercícios para os alunos do ensino fundamental e médio [20]. Além desses, um site que chamou a atenção foi o do Cursinho Etapa [7], que contém uma área de questões dos vestibulares para serem respondidas on-line. Foram encontrados outros colégios que também oferecem ferramentas de ensino on-line, porém não se destacaram tanto quanto esses.

7. CONCLUSÕES

O Ensino baseado na Web ainda está engatinhando, existem poucas coisas definidas. Cada ambiente é implementado de sua própria maneira e os estudos realizados apenas se interessam sobre o ensino totalmente a distância.

O acompanhamento presencial do aluno, fora da sala de aula, também é muito importante, pois estreita o relacionamento entre o professor e o aluno, fazendo com que este se sinta mais motivado a questionar, a tentar, a não ter medo de errar as questões.

Além do que, o aluno que tem os exercícios para fazer em casa, só vai perceber as dúvidas, conseqüentemente em casa. Provavelmente, sem ninguém para ajudar, irá abandonar o exercício e deixar para descobrir sua resposta quando o professor respondê-la, ou copiando de outro aluno.

Com um Ambiente de apoio ao Processo de ensino-aprendizagem mediado por computador como atividade complementar ao ensino presencial, o aluno pode tirar a sua dúvida por e-mail, na hora em que estiver resolvendo os seus exercícios em casa ou usando uma ferramenta de comunicação síncrona (chat). Caso ele use um fórum de discussão, sua dúvida ficará exposta para todos os alunos do mesmo curso, assim, eles poderão debater e também comentar sobre resoluções diferentes de exercícios.

Agradecimentos

Agradecemos ao Programa de Qualificação Docente da Pró-Reitoria de Pós-Graduação da Universidade Cruzeiro do Sul, ao Departamento de Informática, ao NEAD - Núcleo de Educação a Distância e ao CNPq.

Referências

[1]. Alves, R.M; Errico, L and Mesquita, R.C.. SAFES: Um Servidor de Avaliações On-line para Ensino via Web. Belo Horizonte: UFMG. 2003.

[2]. Américo, Colégio S. Disponível em: http://www.csasp.g12.br. Acesso em: 01.out.2003.

[3]. Araújo Jr., C.F.; Silveira, I.F., Amaral, L.H.; Schimiguel, J.; Oliveira, I.C.A. e Turine, M.A.S.T.. Projeto WEL (Web Engineering for Learning) – Um Framework para Desenvolvimento de Objetos de Aprendizagem. *In: Proc. Online Educa Barcelona'2003*, Barcelona, Espanha. 2003.

[4]. Centro de Educação a Distância. Guia de Orientação ao Docente. Biguaçu: UNIVALI. 2003.

[5]. Barboza Jr., A.T.. e-Class Homepage. Disponível em: http://www.unicsul.br/informatica. Acesso em: 10.abr.2004. 2004.

[6]. Delphi, Free Team. Glossário de Informática. Disponível em: http://www.fteam.com/glossario/informatica.shtml. Acesso em: 11.out.2003.

[7]. Etapa, Vestibulares. Desafios ao Raciocínio. Disponível em: http://www.etapa.com.br/desafios/des00_inicio.htm. Acesso em: 01.out.2003.

[8]. Floresta, Colégio. Disponível em: http://www.colegiofloresta.com.br. Acesso em: 10 out. 2003.

[9]. Infoexame, Revista. Guia do Webmaster, Editora Abril. 2002.

[10]. Magazine, Security. Glossário de Informática. Disponível em: http://www.securitymagazine.com.br/glossario.htm. Acesso em: 11.out.2003.

[11]. Musa, D.L.. Um Sistema de Alertas Inteligentes para Ambientes de Ensino na Internet. Porto Alegre: UFRGS. Dissertação de Mestrado. 2002.

[12]. Nobel, Colégio. Disponível em: http://www.colegionobel.com.br. Acesso em: 01.out.2003.

[13]. Objetivo, Colégio. Ensino Médio. Disponível em http://www.objetivo.br/central_ensino_medio/index.htm. Acesso em: 01.out.2003.

[14]. Rudy's, Glossário de Informática. Disponível em: http://www.startpoint.com.br/glossar.htm. Acesso em: 11.out.2003. Biographies.

[15]. Schimiguel, J.; Amaral, L.H.; Araújo Jr, C.F.; Silveira, I.F.; Oliveira, I.C.A.; Ledón, M. and Turine, M.A.S.. Simulações como um recurso para a Educação a Distância. *Conferência Ibero-Americana IADIS WWW/Internet*, Algarve, Portugal, novembro-2003.

[16]. Silveira, I.F.; Araújo Jr., C.F.; Amaral, L.H., Oliveira, I.C.A.. Development of Reusable Learning Objects. *In:Proc. ITHET 2003 – 4th International Conference on Information Technology-Based Higher Education and Training* , Marrakech, Marrocos. 2003.

[17]. Silveira, I.F.; Araújo Jr., C.F.; Oliveira, I.C.A.. e-valuation: A Avaliação Eletrônica como Instrumento de Acompanhamento de Aprendizagem. *In: Proceedings of ICECE'2003 – International Conference on Engineering and Computer Education*, São Vicente, Santos, SP, março-2003.

[18]. Silveira, I.F.; Oliveira, I.C.A.; Schimiguel, J.. Estratégias de Avaliação Eletrônica em Processos de Ensino-Aprendizagem Mediados por Computador. *In: Proceedings of WCETE'2004 – World Congress on Engineering and Technology Education*, Santos, SP, março-2004.

[19]. Tarefanet. Ambiente de Exercícios On-line. Disponível em: http://www.objetivo.br/tarefanet. Acesso em: 01.out.2003.

[20]. Vahl, J.C.J.. Avaliação de um Modelo Computacional para Unidades Educacionais Multimídia. Pelotas: UFPel. Monografia. 2000.

[21]. Wiley, D.A.. Connecting learning objects to instructional design theory: a definition, a metaphor, end a taxonomy. In: D. A. Wiley (Ed.). The instructional use of learning objects. Bloomington. In: Association for Educational Communications and Technology. 2000.

Una herramienta de apoyo en la enseñanza de Geometría Computacional

María Teresa Taranilla
Departamento de Informática, Universidad Nacional de San Luis, Argentina
tarani@unsl.edu.ar

Edilma Olinda Gagliardi
Departamento de Informática, Universidad Nacional de San Luis, Argentina
oli@unsl.edu.ar

Gregorio Hernández Peñalver
Facultad de Informática, Departamento de Matemática Aplicada,
Universidad Politécnica de Madrid, España
gregorio@fi.upm.es

Resumen

La Geometría Computacional es una disciplina que brinda un marco teórico y formal para dar soluciones a problemas de tipo geométrico. En este sentido, las operaciones entre polígonos brindan soluciones a una gama de aplicaciones del mundo real. Una de estas operaciones de gran utilidad es la denominada Sumas de Minkowski.
Esta operación está definida del siguiente modo: Dados dos conjuntos P y Q $\subset \mathbf{R}^2$, la suma de Minkowski de P y Q, denotada por P $\oplus$ Q se define como P $\oplus$ Q = { p + q : p $\in$ P, q $\in$ Q }.
Respecto de la enseñanza de tópicos generales de Geometría Computacional vinculados a las Sumas de Minkowski, se busca cubrir un núcleo básico en los aspectos teóricos y prácticos lo suficientemente amplio de forma tal que el alumno reciba una visión comprensiva de los temas. En este trabajo se presenta una herramienta de apoyo educativo para el cálculo y la visualización de sumas de Minkowski entre polígonos. Mostramos sus características, destacando sus principales componentes y utilidades.

Palabras claves: Geometría Computacional, Sumas de Minkowski. Operaciones entre Polígonos.

Abstract

Computational Geometry is a discipline which offers a formal and theoretical framework to provide solutions to geometric problems. In this sense, the operation between polygons gives solutions to a wide range of real world applications. One of these operations is known as Minkowski Sum. This operation is defined in the following way: Let P and Q be two sets in $\mathbf{R}^2$, the Minkowski sum of P and Q, denoted by P $\oplus$ Q is the set { p + q : p $\in$ P, q $\in$ Q}.
When teaching Computational Geometry related to Minkowski sum, basic knowledge concerning the practical and theoretical aspects are required in order for student to receive a comprehensive view of these topics. In this paper we describe an educational software tool capable of calculating and displaying graphically the Minkowski sum between polygons. We show the tool emphasizing its main components and utilities.

Keywords: Computational Geometry, Minkowski Sum, Operations between Polygons.

1. Introducción

Muchas disciplinas requieren construir y manejar eficientemente objetos geométricos. Entre ellas se pueden destacar: robótica, visión artificial, computación grafica, sistemas de información geográfica, diseño asistido por computadora, etc. En este sentido, la Geometría Computacional es una disciplina que estudia los problemas desde un punto de vista geométrico, dedicándose al diseño y análisis de algoritmos y estructuras de datos adecuadas para su resolución. Una tarea fundamental de la Geometría Computacional es la identificación de conceptos, propiedades y técnicas que ayuden al desarrollo de algoritmos eficientes. La disciplina tiene su origen en el campo de las aplicaciones, donde disponer de un algoritmo eficiente es, a veces, esencial para poder resolver en tiempo real los problemas planteados. Algunas clases particulares de problemas incluyen la búsqueda y recuperación de objetos geométricos, descomposición geométrica, la aproximación de formas y temas relacionados a proximidad, intersección y visibilidad entre objetos geométricos [1, 3, 11, 12].

Uno de los objetos geométricos con los que frecuentemente se trabaja son los polígonos, ya que constituyen una representación adecuada para la mayoría de los objetos del mundo real y, además, las operaciones que pueden realizarse entre ellos brindan adecuadas soluciones en una variada gama de aplicaciones.

Una de las operaciones que se pueden realizar entre polígonos, es la denominada Suma de Minkowski. Para definir las Sumas de Minkowski, decimos que dados dos conjuntos P y Q $\subset \mathbf{R}^2$, la suma de Minkowski de P y Q, denotada por $P \oplus Q$, es $P \oplus Q = \{p + q: p \in P, q \in Q\}$, donde p + q es un vector que representa la suma de los vectores p y q. Es decir que dados los puntos p = (px, py) y q = (qx, qy), tenemos que p + q = (px + qx, py + qy). Esta operación resulta de gran utilidad en aplicaciones tales como planificación de movimientos de robots, procesamiento de imágenes, sistemas de información geográfica, marcado y corte de moldes, entre otras [2, 5, 7, 8, 9].

Nuestro propósito fue el desarrollo de una herramienta de aplicación que sea utilizada como herramienta de trabajo, de simulación y especialmente de apoyo en la enseñanza de la Geometría Computacional. Dicha herramienta implementa la Suma de Minkowski entre distintos tipos de polígonos, tanto entre polígonos convexos y no convexos. En la enseñanza de tópicos generales de Geometría Computacional vinculados a las Sumas de Minkowski, se cubre un núcleo básico de conceptos teóricos y prácticos, de forma tal que el alumno recibe una visión amplia y comprensiva de los temas.

Con respecto a las herramientas con fines educativos relacionadas a la Geometría Computacional, hay bastante trabajo realizado. Aunque, en el caso particular de las Sumas de Minkowski, no existen herramientas con las características de la que hemos implementado. Esta herramienta forma parte de una colección de trabajos que desde 1988 se vienen realizando en el Departamento de Matemática Aplicada de la Universidad Politécnica de Madrid. Como resultado de estos trabajos se han obtenido una colección de aplicaciones que implementan algoritmos sobre distintos problemas dentro del campo de la Geometría Computacional, tales como cierres convexos y sus aplicaciones, problemas de vigilancia e iluminación, diagramas de Voronoi, entre otros.

Este tipo de aplicaciones son ampliamente usadas como apoyo en la enseñanza de la disciplina. La utilización de medios audiovisuales y de software adecuados tiende a facilitar el proceso de enseñanza. Además, cuando los algoritmos geométricos son mostrados con detalle y debidamente discutidos, se favorece el aprendizaje del alumno.

2. La Herramienta

En los últimos años, Internet y el servicio de Word Wide Web (WWW) se han convertido en un importante medio de comunicación e información y han contribuido a introducir nuevas posibilidades en el área de la educación.
En el ámbito de la Geometría Computacional es posible encontrar muchas herramientas que tienen como objetivo auxiliar y motivar a los alumnos en el proceso de aprendizaje como así también ser usadas como complemento en clases presenciales ayudando a los docentes a impartir conceptos que tradicionalmente eran enseñados en forma teórica y con el uso de elementos didácticos más rudimentarios (pizarrón, filminas etc.).
La presencia de este tipo de herramientas disponibles en Internet, beneficia a la enseñanza de la Geometría Computacional puesto que pueden utilizarse haciendo un real aprovechamiento de los recursos de hardware y software disponibles. Otra ventaja a considerar es la independencia del alumno en su formación respecto de la utilización de plataformas de hardware o software específicas.

Con esto destacamos la importancia de considerar la portabilidad de este tipo de software, tal como se ha considerado en este caso, ya que la herramienta implementada es un *applet* de Java, por lo que no es necesario ningún hardware particular ni tampoco realizar ningún tipo de instalación. Tan sólo es imprescindible tener un navegador instalado y configurado para ejecutar código Java contando con el ambiente de ejecución Java JRE versión 1.4 o superior. Dicha versión puede ser descargada gratuitamente desde el sitio de Sun Microsystems en la siguiente dirección http://java.sun.com/j2se/1.4/download.html .
La herramienta desarrollada es interactiva y visual, fácil de entender, usar y permite mostrar en detalle el funcionamiento de los algoritmos usados para el cálculo de la suma de Minkowski, de modo que tanto el problema propuesto como la solución obtenida puedan apreciarse claramente.

La interacción del usuario con la herramienta es sencilla y rápida. Todas las operaciones, tales como inserción de polígonos, elección del tipo de algoritmo a aplicar para el cálculo de la suma, manejo de las operaciones que se realizan durante la ejecución con demora etc. se pueden realizar fácilmente con el mouse.

La herramienta está enmarcada en un conjunto de páginas web. A continuación se describe brevemente el contenido de cada una de las páginas que componen el entorno de la herramienta:

- *Inicio*: Esta página contiene la presentación, título, autores.
- *Aspectos teóricos*: Contiene información teórica en la que se basa la herramienta.
- *Herramienta:* Muestra el programa que implementa la suma de Minkowski entre polígonos.
- *Ayuda*: Explica el funcionamiento de la herramienta (indica sus componentes y cómo utilizarlos). Se abre en una nueva ventana para poder consultarla mientras se utiliza la herramienta.
- *Enlaces de interés:* Desde aquí es posible acceder a enlaces de interés relacionadas con la Geometría Computacional y a las referencias bibliográficas.

2.1 Descripción de la Herramienta

En el ambiente de la herramienta se distinguen claramente tres zonas: una zona media o zona gráfica, una zona inferior y una zona superior, como se puede apreciar en la figura 1.

.

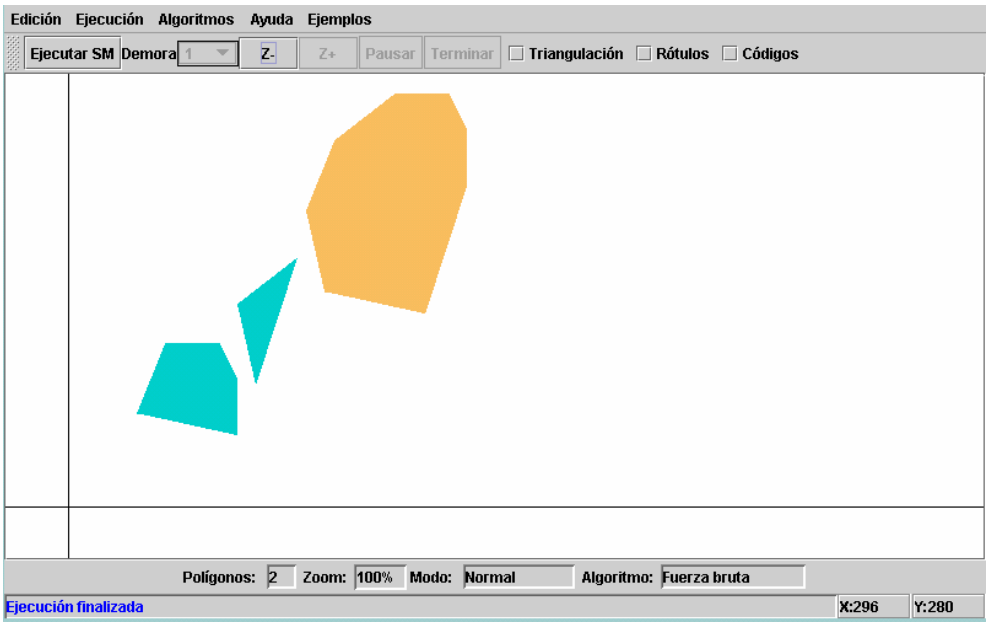


Figura 1: Pantalla del Ambiente

La zona media o zona gráfica es el área donde se dibujan los polígonos y se muestran los resultados de la suma de Minkowski. Puede elegirse el color de fondo de esta zona; las opciones son fondo blanco o fondo negro.

La zona inferior contiene una barra de estado y de información, mostrando en ellas datos tales como la ubicación del cursor del mouse sobre el plano xy, el algoritmo utilizado, información de zoom y, además contiene una Barra de Mensajes donde se muestra información referida al contenido de la zona gráfica y de la ejecución en sí.

En la zona superior del ambiente se encuentran la barra de menúes desplegables y la barra de herramientas, las cuales describimos a continuación.

Barra de menúes

La barra de menúes contiene las siguientes opciones con submenúes desplegables.

Menú Edición

Desde este menú se manejan todas las funciones referentes a la Edición, como se muestra en la figura 2. .

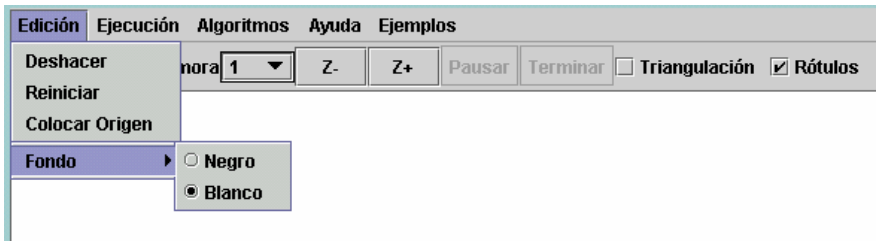


Figura 2: Menú edición

- *Deshacer*: esta opción permite eliminar el último polígono ingresado.
- *Reiniciar*: esta opción borra todo el contenido de la zona gráfica preparando la herramienta para iniciar la resolución de un nuevo problema.
- *Colocar origen:* esta opción permite ubicar el origen de las coordenadas en cualquier posición de la zona gráfica.
- *Fondo:* esta opción permite cambiar el color de fondo de la zona gráfica, pudiendo elegir entre los dos colores disponibles, blanco o negro.

Menú Ejecutar

Desde este menú, se ejecuta el cálculo de la Suma de Minkowski y se selecciona el modo de ejecución con el que se calculará la misma. Se permite al usuario elegir entre dos modos de ejecución, continuo o con demora, como se aprecia en la figura 3.

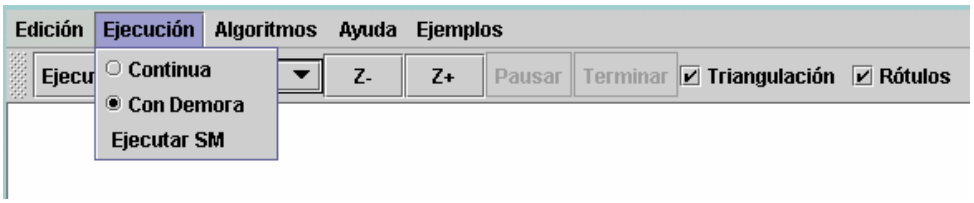


Figura 3: Menú Ejecución

Menú Algoritmo

Desde este menú es posible elegir el algoritmo que se usará para el cálculo de la suma de Minkowski. En esta herramienta se han implementado dos algoritmos: el algoritmo de Fuerza Bruta y el algoritmo Mejorado. La figura 4 muestra este menú.

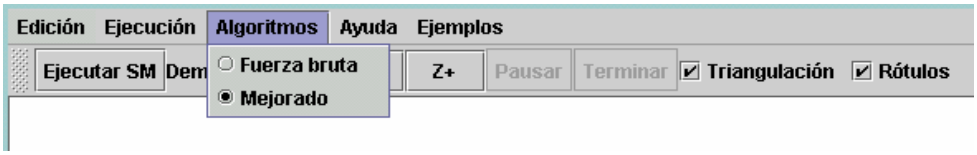


Figura 4: Menú Algoritmos

Menú Ayuda

Desde este menú se accede a la ayuda disponible y a la información acerca de los autores, como se observa en la figura 5.

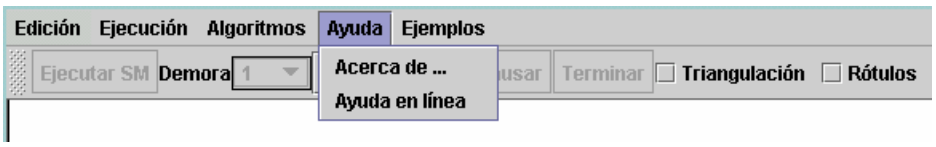


Figura 5: Menú Ayuda

Menú Ejemplos

Desde este menú es posible elegir diferentes ejemplos de problemas que calculan la suma de Minkowski entre distintos tipos de polígonos.

Barra de Herramientas

La Barra de Herramientas ubicada en la zona superior del ambiente contiene:

- **Botones de acciones**
  - *Botón de ejecución:* este botón denominado Ejecutar SM permite iniciar la ejecución del cálculo de la suma de Minkowski.
  - *Botones de zoom:* Los botones Z+ y Z- se utilizan para "acercar" o para "alejar" la vista del contenido de la zona gráfica y ver un porcentaje mayor del contenido a tamaño reducido.
  - *Botón Pausar/Continuar:* si la opción elegida para la ejecución es con demora, este botón permite manejar la ejecución haciendo pausas cuando el usuario así lo desee y luego continuar la ejecución.
  - *Botón Terminar:* este botón permite finalizar una ejecución con demora desde cualquier punto en que ésta se encuentre, obviando la demora.
- **Opciones**
  - *Demora:* permite elegir el tiempo de demora (en segundos) entre cada paso para el modo de ejecución con demora.
  - *Triangulación*: permite activar y desactivar la opción de mostrar la triangulación realizada en los polígonos no convexos.
  - *Rótulos:* permite activar y desactivar la opción de mostrar las coordenadas de los vértices de los polígonos ingresados en la zona gráfica.
  - *Códigos:* permite activar y desactivar las opción de mostrar en una ventana auxiliar el código del algoritmo que se esta ejecutando, cuando el tipo de ejecución elegida es con demora.

2.2 Funcionalidad de la Herramienta

La herramienta permite fácilmente dibujar polígonos simples tanto convexos como no convexos y calcular la suma de Minkowski entre ellos. A medida que se ingresan los polígonos se realizan controles para que no sea posible el ingreso de polígonos que no sean simples. Además en el caso que el polígono ingresado sea no convexo, inmediatamente después de que se termina de dibujar, es triangulado, mostrándose la triangulación obtenida en pantalla si el usuario así lo desea. En la figura 6, se observan dos polígonos no convexos y su correspondiente triangulación.

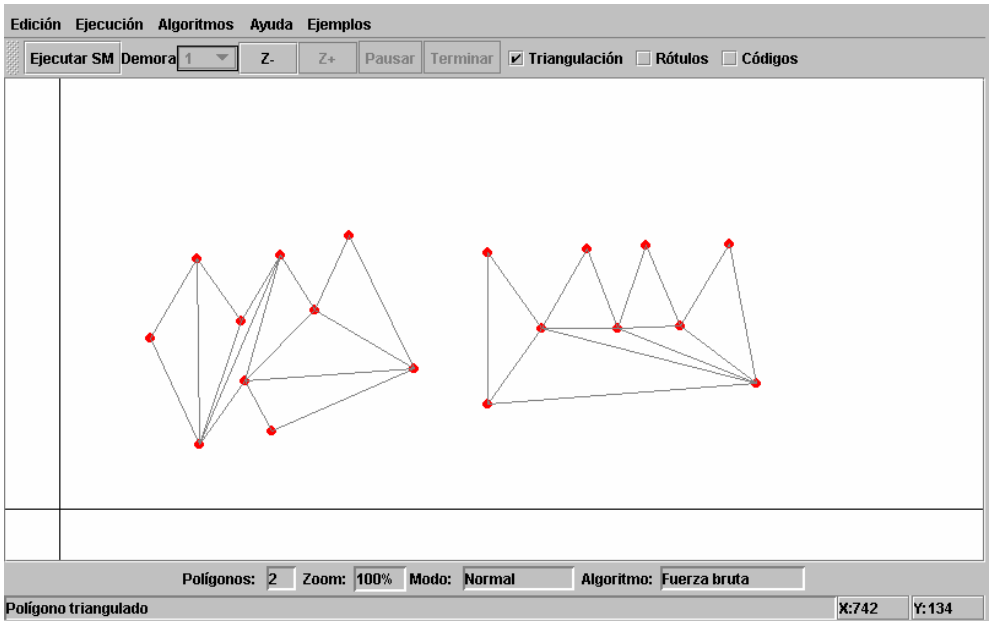


Figura 6: Polígonos no convexos triangulados

Se permite dibujar polígonos en cualquier cuadrante del plano *xy*, creando los ejes de coordenadas y permitiendo al usuario ubicarlos y moverlos a cualquier punto de la herramienta.

Para calcular la suma de Minkowski deben ingresarse dos polígonos. En el caso de ingresar más de dos polígonos, el resultado será la suma de Minkowski de cada uno de los polígonos con el primer polígono ingresado.

Una vez ingresados dos o más polígonos, la herramienta permite realizar la suma de Minkowski, teniendo como opciones dos tipos de ejecución: **ejecución continua** o **ejecución con demora**. El modo de ejecución puede elegirse en el menú ejecución de la barra de menúes.

La ejecución continua muestra directamente la solución final al problema planteado, en cambio la ejecución con demora permite observar los pasos de la resolución del problema con la demora elegida por el usuario. La demora se determina desde la lista desplegable a la derecha del rótulo *Demora* ubicado en la barra de herramientas. La demora por defecto es de un segundo. Es posible cambiar el tiempo de la demora durante la ejecución, eligiendo de dicha lista otro intervalo de tiempo de demora.

Cuando la opción de ejecución elegida es con demora es posible detener la ejecución en cualquiera de los pasos que se encuentre y luego continuar con la ejecución desde el punto en que se detuvo. También, es posible finalizar la ejecución con demora, desde cualquier punto en que ésta se encuentre, obviando la demora.

Una opción interesante que puede elegir el usuario en una ejecución con demora es ver el pseudocódigo del algoritmo en ejecución y seguir junto con la ejecución visual las líneas de código que se esta ejecutando en ese momento, permitiendo de esta manera un entendimiento más profundo aún de los algoritmos que se utilizan.

Otra característica de la herramienta es que, como el resultado de la suma de Minkowski entre dos polígonos puede estar fuera del alcance visual de la herramienta, se provee la opción de alejar y/o acercar la visión del plano. Esta operación denominada *zoom* está disponible en la barra de herramientas.

Es válido aclarar que durante la ejecución con demora es posible cambiar el tiempo de demora, cambiar el porcentaje de *zoom*, mover los ejes de coordenadas y otras opciones sin ningún inconveniente.

2.3 Aspectos de la implementación

Para el desarrollo de la herramienta implementada elegimos el lenguaje **Java**. La elección del lenguaje de programación quedó determinada, en gran parte, por el objetivo de la herramienta, por el uso que se pretende hacer de ella y para aprovechar las ventajas que ofrece en la actualidad el acceso generalizado a Internet. Es decir, el código del programa puede estar localizado en un servidor pero puede utilizarse desde cualquier lugar, sin necesidad de más ayuda que la del navegador Web que se utilice habitualmente y en forma independiente de la plataforma que el usuario esté utilizando.

Uno de los objetivos es que nuestra herramienta se ejecute en forma independiente de la plataforma y sin que sea necesaria una instalación previa. Para ello, en lugar de implementar una aplicación standard Java decidimos implementar lo que se denomina applet en Java.

Una aplicación es un programa escrito en Java al que no le falta nada, es independiente y puede ejecutarse emitiendo una orden desde la interfaz de usuario del sistema operativo subyacente. Para ejecutar una aplicación Java es necesario tener instalado en el sistema una máquina virtual Java y haber instalado previamente la aplicación.

Para poder ejecutar un applet es necesario sólo poseer un navegador Web con capacidades Java. Un applet debe ejecutarse dentro de un navegador Web y no puede ser ejecutado en forma independiente. Cuando un navegador Web carga una página Web que contiene un applet, el navegador descarga el código del applet desde el servidor Web y lo ejecuta en el sistema local. Las ventajas principales del applet son que no necesita instalación en el sistema local y que la interfaz de usuario está dada por el navegador web.

Antes de iniciar el cálculo de la suma de Minkowski, es posible elegir el algoritmo que se desea usar en la resolución del problema. Para esta herramienta se implementaron dos algoritmos que calculan la suma de Minkowski de polígonos convexos, el algoritmo que denominamos de *algoritmo de fuerza bruta*, que tiene una complejidad de ejecución de orden cuadrático y el algoritmo que llamamos *algoritmo mejorado*, con una complejidad de ejecución de orden lineal.

La idea de implementar los dos algoritmos es que se puedan comparar la ejecución de los mismos; esto se hace así principalmente por los fines educativos que persigue la herramienta. De esta forma, es posible apreciar los distintos costos de ejecución que tienen los algoritmos, y observar con detenimiento cómo por diversas estrategias, se arriba a la solución de un problema.

Para llevar a cabo la implementación de los algoritmos que calculan la Suma de Minkowski fue necesaria la implementación de otros algoritmos propios de la Geometría Computacional. En el algoritmo de fuerza bruta para el cálculo de la suma de Minkowski usamos el conocido Scan de Graham, que permite calcular el cierre convexo de una nube de puntos con $O(n \log n)$ [4]. En el algoritmo mejorado, usamos una variación especial del método de rotación de calibres para calcular los puntos extremos de los polígonos [10]. Otro algoritmo auxiliar que utilizamos es la triangulación de polígonos, aplicado a los polígonos no convexos. En este caso implementamos el algoritmo de Kong, que es una variación del algoritmo de "corte de orejas" [6].

3. Conclusiones

En este trabajo se describió una herramienta didáctica para ser utilizada como apoyo en la enseñanza de la Geometría Computacional, permitiendo mostrar el funcionamiento de los algoritmos usados para el cálculo de la suma de Minkowski entre polígonos.

La utilización de software adecuado facilita el proceso de enseñanza y favorece al alumno en el proceso de aprendizaje. Consideramos que este tipo de herramientas ofrecen facilidades en la enseñanza de la Geometría Computacional, puesto que las prácticas de laboratorios constituyen un complemento importante para afianzar conceptos teóricos que sirven de marco apropiado en diferentes aplicaciones que se pueden modelar como problemas de tipo geométrico. Las implementaciones que permiten mostrar con detalle la ejecución de distintos algoritmos que realizan la misma tarea, en forma pausada, continua o con demora, son útiles para mejorar el entendimiento de diversos algoritmos de cualquier temática.

La herramienta está disponible en la siguiente dirección:

http://www.dma.fi.upm.es/docencia/segundociclo/geomcomp/aplicaciones.html

Cabe destacar que, esta herramienta puede ser ampliada para realizar otro tipo de operaciones algebraicas entre diferentes tipos de polígonos, así como también se puede pensar en el desarrollo de otras herramientas que implementen sumas de Minkowski entre conjuntos de polígonos y sumas de Minkowski en tres dimensiones.

Referencias

[1] Abellanas Oar, M. *Descubriendo la Geometría Algorítmica*, 2000.

http://www.dma.fi.upm.es/mabellanas/divulgación/GeometriaAlgoritmica.html

[2] Agarwal, P.K.; Flato, E.; Halperin, D., *Polygon Decomposition for efficient construction of Minkowski sums*, Computational Geometry: Theory and applications Nº 21, (39-61), 2001.

[3] de Berg, Kreveld, Overmars, Schwarzkopf. *Computational Geometry: algorithms and applications*, Springer Verlag, BH 1997

[4] Graham, R.L. *An efficient algorithm for determining the convex hull of a finite planar set*, Information Process Letters, 1:132-133, 1972

[5] Heywood I., Cornelius S., Carver S., *Geographical Information Systems*, Addison-Wesley Longman, New York, 1998.

[6] Kong, X.; Everett, H.; Toussaint, G. T., *The Graham scan triangulates simple polygons*, Pattern Recognition Letters, vol. 11, November 1990, pp. 713-716.

[7] Latombe, J.C., *Robot Motion Planning*, Kluwer Academic Publisher, Boston, MA, 1991.

[8] Li, Zhenyu *Compaction Algorithms for Non-Convex Polygons and their Applications*, tesis doctoral, Universidad de Harvard, 1994

[9] Serra J., *Image Analysis and Mathematical Morphology*, Academic Press, New York, 1982.

[10] Toussaint, G.T. *Solving geometric problems with Rotating Calipers*, Proceedings of IEEE MELECON'83, Athens, Greece, May 1983, pp. A10.02/1-4.

[11] Toussaint, G.T. *Computational Geometry*, Edited by North-Holland, Amsterdam, 1985 .

[12] Toussaint, G.T. *What is computational geometry?* Proc. IEEE, vol. 80, No. 9,. (1347-1363),1992.

Utilização de um Sistema ERP no Apoio às Atividades de Ensino na Unisul

Allan Augusto Platt
Universidade do Sul de Santa Catarina, Administração
Florianópolis, Santa Catarina, Brasil, 370
allan@unisul.br

Ricardo Vilarroel Dávalos
Universidade do Sul de Santa Catarina, Sistemas de Informação
Florianópolis, Santa Catarina, Brasil, 370
rdavalos@unisul.br

Lia Caetano Bastos
Universidade Federal de Santa Catarina, CTC - ECV
Florianópolis, Santa Catarina, Brasil, 88040-900
lia@ecv.ufsc.br

Abstract

This article presents an including model that proposes the use of a Enterprise Resources Planning (ERP) to support the teaching-learning processes in the courses of the areas of business and technology in four multi-campus of the University of Santa Catarina's South - Unisul. This university has been implanting the SAP R/3 system to support its administrative activities and it has been creating a research group to establish forms of to introduce the ERP in class room and to support to the pedagogic projects of the different courses. The proposed model is based on experiences accomplished in another universities and it uses available resources of the system, so much for employment in the disciplines, as for integration among several disciplines of the different courses and multi-campus of the university.
Keywords: Teaching/Learning Strategies, Information Systems, Information Technology.

Resumo

Este artigo apresenta um modelo abrangente que propõe a utilização de um Sistema Integrado de Gestão (*Enterprise Resources Planning* – ERP) para apoiar os processos de ensino-aprendizagem nos cursos das áreas de negócios e tecnologia em quatro campi da Universidade do Sul de Santa Catarina - Unisul. Esta universidade implantou o sistema SAP R/3 para apoiar suas atividades administrativas e criou um grupo de pesquisa para estabelecer formas de introduzir o sistema ERP em sala de aula e apoiar aos projetos pedagógicos dos diferentes cursos. O modelo proposto está baseado em experiências realizadas em outras universidades e utiliza recursos disponíveis do sistema, tanto para emprego nas disciplinas, como para integração interdisciplinar dos diferentes cursos e campi da universidade.
Palavras chave: Estratégias de Ensino/Aprendizagem, Sistemas de Informação, Tecnologia de Informação.

1. INTRODUÇÃO

Com o avanço da Tecnologia de Informação (TI), as empresas tornam-se cada vez mais dependentes da informação bem como de sistemas computacionais mais sofisticados para o suporte de suas atividades. Hoje, mais do que nunca, a informação significa poder, possibilitando a busca de diferenciais competitivos, melhor atendimento aos clientes e a otimização da cadeia de suprimentos. Deter e controlar a informação de modo a reagir rapidamente a uma exigência do mercado é uma necessidade que nenhuma empresa pode ignorar [1].

As empresas que estão mudando usam a tecnologia como um instrumento para obtenção de competitividade no desenvolvimento de novos produtos e serviços, para forjar novos relacionamentos com os fornecedores, para se transformarem em empresas de ponta em relação a seus competidores, ou mudar radicalmente suas operações internas ou estrutura.

Sob estas premissas, em meados da década passada, surge o conceito de Sistemas Integrados de Gestão *(Enterprise Resources Planning* - ERP), tecnologia capaz de organizar e integrar as informações armazenadas nos computadores de uma organização, de forma a eliminar dados redundantes ou desnecessários, racionalizar processos e distribuir a informação de forma estruturada, fidedigna e em tempo real, para todas as áreas de uma empresa. O ERP pode ser entendido como a espinha dorsal da TI na empresa, dentro da filosofia de centralizar a complexidade e distribuir a informação.

Uma das conseqüências do uso cada vez maior desta tecnologia é a demanda por mão-de-obra capaz de entender, utilizar, configurar, programar, desenvolver soluções específicas, enfim desvendar o ERP. Outra carência concerne a própria formação técnica dos usuários, realizada por empresas de implantação de ERP ou até mesmo pelos fornecedores destes sistemas. O entendimento científico acaba não sendo relevado pelas empresas ao treinarem seus funcionários.

Esta lacuna por mão-de-obra com conhecimentos holísticos envolvendo sistemas de informação, ERPs, suas funcionalidades, arquitetura tecnológica e a visão empresarial só recentemente têm recebido a devida atenção do meio acadêmico, seja no estudo do sistema, seja na formação de pessoal com mais gabarito para atender as demandas das empresas adquirentes do ERP, das empresas de consultoria e das próprias fornecedoras desta ferramenta.

As universidades, conscientizadas da importância do assunto, têm celebrado acordos com fornecedores de *hardware* e *software*, recebendo recursos geralmente a custos simbólicos. Professores são treinados, laboratórios equipados com o ERP passando a ser tema de grande importância, a ponto de gerar alterações curriculares (Ex.: Louisiana State University, California State University, Massachusett Institute of Technology, Universidade de São Paulo, Universidade Newton Paiva, etc).

A Universidade do Sul de Santa Catarina - Unisul têm se tornado um grande exemplo (*case*) que corrobora tanto a necessidade de uso de ferramentas de TI no auxílio ao gerenciamento de seu crescimento, com a implementação do ERP R/3, da empresa alemã SAP, em suas atividades administrativas, como na formação de mão-de-obra qualificada ao constituir um grupo de pesquisa que possa inserir o ERP como ferramenta de ensino-aprendizagem em seus cursos.

Apesar de inicialmente ter sido estabelecido um convênio com a SAP também para a parceria acadêmica, vários problemas ocorreram, levando a troca de parceria para a empresa brasileira Microsiga. A partir da constatação destes problemas, verificou-se a necessidade de uma metodologia que pudesse orientar seus agentes a atingir seus objetivos de inclusão de sistemas ERP nos projetos pedagógicos dos diferentes cursos.

Assim, o objetivo principal deste artigo é propor um modelo abrangente que considere a utilização de um sistema ERP para apoiar o processo de ensino-aprendizagem nos cursos das áreas de negócios e tecnologia em quatro campi da Unisul.

2. SISTEMAS INTEGRADOS DE GESTÃO

O desenvolvimento de TI permitiu que as organizações passassem a utilizar Sistemas de Informação (SI) para apoiar suas atividades. Vários destes sistemas foram desenvolvidos para atender aos requisitos específicos das diversas unidades de negócio, plantas, departamentos e escritórios. Um SI poderia ser compreendido como um conjunto de componentes inter-relacionados, desenvolvidos para coletar, processar, armazenar e distribuir informações, facilitando a coordenação, o controle, a análise, a visualização e o processo decisório nas organizações [3].

Estas organizações de escala mundial, num esforço para permanecerem competitivas, têm direcionado seus esforços nas últimas duas décadas na melhoria dos processos de negócio. Outro esforço, realizado por empresas de todos os tamanhos visando o alinhamento dos processos de negócio com a TI, tem propiciado o aparecimento e o crescimento dos sistemas ERP. Como resultado, estes sistemas têm apresentado uma abordagem de empresa integrada que é diferente da tradicional [14].

Fatores como a pressão competitiva global em busca das melhores práticas, a promoção da reengenharia de processos de negócio pelas grandes firmas de consultoria, a tendência internacional de privatização dos serviços públicos, a ampliação dos recursos destinados a área de tecnologia da informação para suportar a reengenharia das empresas e a pressão para mudanças rápidas de adequação dos SI, têm propiciado a expansão significativa dos ERP [14].

Os ERPs representam o estágio mais avançado dos sistemas tradicionalmente chamados de MRP II (*Manufacturing Resource Planning*), projetados para o planejamento e controle de recursos produtivos. Estes sistemas por sua vez evoluíram dos sistemas MRP *(Material Requirement Planning*) e BOM (*Bill of Materials*) que abrangiam funções mais restritas do setor de suprimentos da indústria [4].
É composto basicamente de módulos que atendem as necessidades de informação, ligados a todos os processos operacionais, produtivos, administrativos e comerciais. A Figura 1 ilustra a estrutura típica de funcionamento integrado de um ERP, a partir de uma base de dados única [5].

Figura 1 - Estrutura típica de funcionamento de um ERP.

Vários autores caracterizam o ERP como [1] e [14]:

- orientados à processos, substituindo a estrutura funcional dos sistemas legados;
- multifuncionais, incorporando processos de compras, vendas, finanças, etc;
- integrados (base de dados única), ou seja, a medida que um dado é registrado numa das funcionalidades, todas as demais que se relacionam com esta recebem este registro;
- de estrutura modular, podendo ser utilizados com qualquer combinação de módulos, ou apenas parte deles. Outra possibilidade é a inclusão de sistemas específicos para determinadas indústrias, junto ao ERP.
- flexíveis para responder as necessidades de mudança de uma empresa através da tecnologia cliente/servidor, da conectividade de base de dados aberta (ODBC), etc;
- meio de possibilitar a expansibilidade na cadeia de suprimentos (interface para parceiros externos, *e-commerce*); e
- meio de sistematização no lançamento das informações, permitindo o controle em tempo real, refletindo sempre a situação atualizada da empresa.

Muitos ERPs são comercializados em pacotes contendo módulos básicos para a gestão do negócio. Módulos adicionais podem ser adquiridos individualmente, em função do interesse e da estratégia da empresa. Todos esses aplicativos são completamente integrados, a fim de propiciar consistência e visibilidade a todas as atividades inerentes aos processos da organização. Nomes comerciais de ERP como SAP, BAAN, *Oracle Aplications*, BPCS, *Peoplesoft*, *JDEdwards*, MFG/Pro, Microsiga, Datasul, dentre outros, passaram a fazer parte das empresas de médio a grande porte no Brasil e no exterior [4].

3. O SISTEMA ERP NO ENSINO SUPERIOR

Desvendar o ERP é o assunto que tem vindo à tona no meio acadêmico, possibilitando que algumas parcerias estejam sendo celebradas entre instituições de ensino superior e as fornecedoras do sistema. No entanto, cada iniciativa deste porte têm sido tratada de maneira peculiar por cada instituição, em função da forma como seus dirigentes e professores têm interpretado a relevância de ter em seus cursos uma ferramenta deste tipo.

Alguns autores colocam que há uma clara oportunidade para as universidades obterem valor agregado e percebido em seus currículos dos cursos voltados aos negócios, engenharia e computação, aumentando a empregabilidade dos recém-formados e elevando a procura por matrículas nestes cursos [7] e[13].

Um ERP é na realidade um modelo empresarial baseado em computador extremamente complexo que opera em tempo real e que pode responder a mudanças em seu ambiente operacional. Se este modelo pode ser trazido ao ambiente pedagógico, poderá ser uma ferramenta muito útil demonstrando a relevância do conteúdo do curso para o mundo real. Esta possibilidade adicionada ao aprendizado prático junto ao sistema através da navegação em seus módulos, na execução de atividades (transações) e na produção de relatórios torna esta iniciativa bastante relevante ao ambiente acadêmico [7].

De acordo com [13], para que o uso de um sistema ERP numa parceria acadêmica tenha sucesso, deve ser desenvolvido e implementado um plano sólido, determinando tempo, esforço do pessoal envolvido e recursos necessários ao desenvolvimento da infraestrutura. Este plano deve estar em concomitância com os objetivos da parceria e que podem ser identificados como:

- desenvolvimento de um simulador computacional;
- exposição dos alunos às situações do mundo real;
- desenvolvimento de um currículo interdisciplinar;
- enriquecimento de currículos específicos com o uso do sistema ERP;
- desenvolvimento de oportunidades de novas pesquisas; e
- criação de vantagem competitiva.

Algumas experiências serão relatadas a seguir como exemplo das várias possibilidades que podem advir de um programa de integração de sistemas ERP nos currículos universitários.

3.1 Experiências da Universidade do Estado de Louisiana

Watson e Schneider [13] apresentam em seu trabalho uma metodologia de aplicação do ERP, resultado da parceria entre a Universidade do Estado de Louisiana e a SAP, que utiliza nos seus cursos vários cenários descritos a seguir:

- *Navegação na Internet* – os alunos acessam uma gama de dados e informações técnicas e de negócio via internet;
- *Navegação na Intranet* – através do uso do sistema de ajuda on-line disponível com o sistema, os alunos aprendem sobre como processar uma transação, trazendo uma compreensão das regras de negócio, definições e recurso dos processos;
- *Execução de transações* – através de um roteiro (*business script*), os alunos executam uma sucessão de tarefas usadas para simular um processo empresarial (processamento de pedidos, por exemplo) ou uma atividade (geração de relatórios e análise);
- *Busca de um Objetivo* – ao fornecer aos estudantes um objetivo (elaborar um relatório para uma filial ou de um canal de distribuição), permiti-se que eles imaginem e descubram o melhor caminho para fazê-lo;
- *Desenvolvimento* - Trabalhar num ambiente de desenvolvimento integrado para definir necessidades e desenvolver soluções usando ferramentas de desenvolvimento é o *modus operandi* neste cenário;
- *Administração do Sistema* - Esta abordagem possibilita que o estudante desenvolva atividades típicas como administrador do sistema, monitorando e produzindo relatórios para os usuários do sistema; e
- *Simulação, Implementação e Atividade de Consultoria* - Este cenário contempla a comparação das melhores práticas e o desenvolvimento de alternativas possíveis de processos específicos da empresa, incluindo análise de *gaps*. O uso de ferramentas de implementação de ERP, customizando um processo ou toda a organização, também fazem parte deste nível de abordagem, como também a avaliação dos guias de usuários desde o início da implementação até as fases finais do processo;

Uma metodologia utilizando módulos de conhecimento, denominados *KnowDules* foi criada para suportar a integração do ERP em seus cursos. Estes K*nowDules* foram desenvolvidos para serem módulos de aprendizagem *on-line*, incluindo conteúdo de apresentação, notas, referências e *links*, além dos exercícios utilizando o ERP. Cada K*nowDule* pode durar de dois a dez encontros (de 80 minutos cada) e está incorporado em cursos específicos, tanto em nível de graduação como de pós-graduação.

3.2 Experiências de Guthrie e Guthrie

Os autores Guthrie e Guthrie [7] citam que há três possibilidades de integração do ERP nos currículos universitários definidos a seguir:

- *Integração inter-modular*: os estudantes são expostos aos vários módulos do ERP em um único curso ou numa série de cursos relacionados a implementação ou manutenção do sistema;
- *Integração modular do currículo*: nesta abordagem os alunos cursam uma disciplina e aprendem os conceitos no módulo do ERP correspondente; e
- *Integração interdisciplinar*: aqui os estudantes acessam o ERP com uma base de dados fictícia, observando e analisando uma empresa sob diferentes pontos de vista, reagindo e adaptando-se as alterações e intervenções realizadas por outros alunos de outras disciplinas.

Cinco níveis de imersão técnica desta iniciativa ERP - Universidades são propostos pelos autores da seguinte forma:

- *Modelo Empresarial*: Neste nível o software é utilizado como ferramenta demonstrativa durante as aulas expositivas. Os estudantes apenas navegam no sistema conhecendo sua composição, a maneira em que é utilizado, sem aprofundarem-se nas suas funcionalidades;
- *Tutorial*: Esta abordagem permite que o aluno desenvolva o auto-aprendizado fora da sala de aula, com o auxílio da internet , consultando o site do fornecedor de ERP ou *cd-rom*;
- *Laboratório de Projeto*: O laboratório de projeto é um ambiente simulado no sistema ERP que permite que os estudantes manuseiem o software. O estudante usa um banco de dados pronto de demonstração e completa uma série de atividades estruturadas que lhe dá, tanto exposição para as atividades de dia-a-dia de uma empresa, como experiência com o ERP;
- *Curso dedicado*: São projetados para ensinar habilidades relacionadas ao ERP, podendo ser incluídos em um programa de certificação formalmente reconhecido. Tanto o conteúdo, assim como os instrutores são fornecidos pelo vendedor do software; e
- *Prática Integrada*: Neste nível os estudantes são agrupados em equipes de projeto ao longo de um semestre vivenciando o ambiente real de uma empresa ou através de uma simulação ou jogo em que o ERP é a ferramenta utilizada.

3.3 A Experiência de Ensino a Distância da Universidade de Victoria

Os autores Hawking e McCarthy [10] abordam em seu trabalho a experiência da Universidade de Victoria (Austrália) no desenvolvimento de um modelo de aprendizado virtual do ERP (*eLearning*), com o intuito de facilitar o ensino em universidades asiáticas, parceiras nesta iniciativa. O modelo descrito a seguir foi desenvolvido e testado em universidades da Malásia e de Hong Kong em 2001. Quatro tecnologias são apresentadas no apoio ao modelo conforme descritas a seguir:

- *Application Service Provision (ASP)*: Esta tecnologia caracteriza-se pelo fornecimento de infraestrutura e suporte à "hospedagem" do sistema, permitindo que os usuários possam acessar o ERP por via remota, através da internet;
- *Sala de aula virtual*: Enquanto a tecnologia ASP possibilita o acesso ao ERP via internet, a Sala de Aula Virtual permite que o aluno assista às aulas enquanto elas estão sendo ministradas, acesse os slides disponibilizados, faça questionamentos e obtenha respostas diretamente do palestrante, caracterizando-se por um serviço de "mão-dupla" que dinamiza o processo de ensino-aprendizagem;
- *Tutor*: Esta ferramenta é usada no desenvolvimento de um material de apoio (arquivo tutorial) interativo num ambiente simulado do ERP. O professor registra as seqüências de ações para a realização de uma transação no sistema, capturando as telas utilizadas neste caminho, as possíveis variações nos passos para a realização da transação, além de inserir comentários e textos descritivos sobre ações realizadas; e
- *Ponto central*: Além de fornecer materiais de aprendizado, funciona também como um portal para as várias ferramentas de ensino disponibilizadas às universidades asiáticas, possibilitando o *download* de arquivos, slides, aulas em vários formatos, tarefas, etc.

3.4 Experiências de ensino de uma funcionalidade do ERP na Universidade de Victoria

Como parte integrante de seu curso de Bacharelado em Negócios em Gestão de Recursos Humanos, foi introduzido o sistema SAP R/3 na disciplina de Sistemas de Informação de Recursos Humanos, envolvendo duas horas semanais de aulas expositivas e uma hora de seminários por 13 semanas, sendo esperado que os alunos aprendam tópicos relacionados a implementação, dados cadastrais de recursos humanos e opções de relatórios.

Os alunos são iniciados no sistema SAP R/3 através de uma série de exercícios que reforçam as técnicas de navegação e acessam a documentação de "*help*" através de vários métodos. Em seguida eles têm contato com o programa mestre de um produto e criam relatórios onde são executados cálculos simples e é demonstrado o conceito de desdobramento da informação. Com estes conceitos básicos dominados, o foco é então direcionado ao módulo de Recursos Humanos [10].

Através de um estudo de caso fictício, os alunos recebem os dados sobre a estrutura de uma empresa com unidades organizacionais, cargos e níveis hierárquicos e registra-os no sistema. Em seguida o processo de recrutamento é elaborado (criação de vagas, elaboração de anúncios, registro dos dados do candidato, cartas de aceite ou rejeição, etc.).

Um benefício adicional na adoção do ERP na academia é relativo ao feedback dado pelas indústrias. Como parte da avaliação, é exigido dos alunos que completem um estudo de caso numa companhia de Recursos Humanos para então apresentar seus resultados em sala. Os autores citam que em muitos casos, as empresas já possuem ERP, inclusive o R/3 da SAP, realçando as habilidades dos alunos no mercado, já que desenvolvem seu aprendizado no mesmo sistema, além das próprias atividades curriculares [10].

3.5 Experiências de Parceria entre as Universidades de Widener (EUA) e Muenster (Alemanha).
Os autores Antonucci e Meuhlen [2] apresentam o trabalho de parceria entre duas universidades (Widener nos EUA e Muenster na Alemanha) utilizando um sistema ERP em conexão com a internet. Para eles, a integração entre as plataformas de negócio apoiadas na internet e o comércio eletrônico introduziu um novo nível de integração nos processos de negócio: a habilidade para desenvolver os processos interorganizacionais.
As fases do desenvolvimento deste modelo de currículo colaborativo incluíram, segundo os autores:

- implementação de um caso de terceirização das atividades de *help-desk* utilizando três empresas;
- acesso pela internet com a troca de mensagens utilizando a linguagem XML;
- uso da tecnologia de *workflow*; e
- integração do processo de B2B.

Esta parceria forneceu aos estudantes a oportunidade de vivenciar uma experiência colaborativa de âmbito internacional, além de adquirirem habilidades em ensino à distância e técnicas de ensino. As atividades desenvolvidas pelos alunos de cada universidade (16 estudantes nos EUA e 8 na Alemanha) incluíam:

- participação em painéis de discussão em tempo real entre as universidades;
- participação em fóruns de discussão na internet com o intuito de resolverem atividades e assuntos relacionados aos processos interorganizacionais;
- análise do processo do cenário de *help-desk* entre as universidades visando o desenvolvimento de *workflows* (fluxos de processos de negócio) independentes que precisavam interagir, simulando e desenvolvendo um estudo de caso B2B (*business to business)*;
- formação de equipes com estudantes das duas universidades participando de um mesmo cenário: um site de *B2B* foi desenvolvido pelos alunos para ser utilizado como fórum colaborativo *on line*.

4. O ERP NA UNISUL

A Universidade do Sul de Santa Catarina - UNISUL, hoje consolidada como Universidade Comunitária Regional, irradia suas ações através de unidades descentralizadas em sua área de abrangência. Sua área de atuação abrange quatro microrregiões do Estado de Santa Catarina, e seus campi, em Tubarão (I), Araranguá (II), Grande Florianópolis (III) e Norte da Ilha (IV), estão localizados nos municípios pólos, conforme apresentados no mapa indicativo da Figura 2.

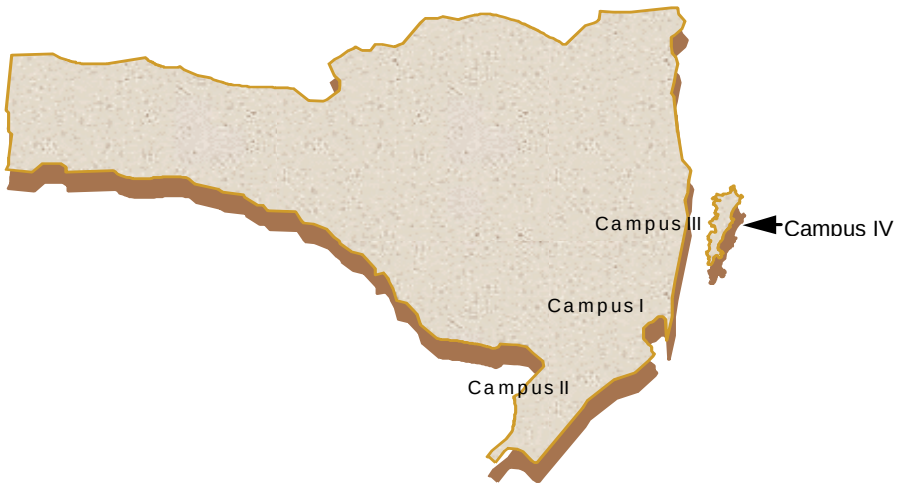


Figura 2 - Localização dos Campi da Unisul no Estado de Santa Catarina.

Na busca de um gerenciamento eficiente das informações, a Unisul implantou o SAP/R3 mediante o "Projeto Visão". O objetivo deste projeto foi de reestruturar a universidade para promover uma melhor integração dos alunos, docentes e colaboradores aos processos de gestão universitária.

A implantação do SAP/R3 contou com o comprometimento da direção e foi realizada por professores e funcionários da universidade, com a participação de consultores designados pelas empresas envolvidas na parceria. A importância da implantação deste sistema se justifica pela capacidade de apoio aos procedimentos de gestão e à tomada de decisões. Como a universidade apresenta uma grande quantidade de dados e processos, existe a necessidade de contar com procedimentos de tratamento da informação mais precisos e rápidos.
Durante a implantação do SAP/R3, foi criado na Unisul o GSIG (Grupo de Sistemas Integrados de Gestão), um grupo estratégico de pesquisa para dar suporte às ações acadêmicas e administrativas na Unisul, integrado por professores e funcionários e liderados pelo Diretor de Pesquisa da Unisul.
Os objetivos iniciais colocados pelo grupo foram os seguintes:

- gerar conhecimentos e tecnologia em sistemas integrados de gestão por processos, para a melhoria da prática acadêmica e administrativa;
- gerar conhecimentos em gestão universitária;
- ser referência como grupo de pesquisa aplicada em sistemas integrados de gestão;
- ser um grupo de pesquisa de referência em gestão, gerando conhecimento para o alto desempenho de uma instituição de ensino superior; e
- gerar conhecimento na área dos sistemas integrados ERP e posterior repasse dos conhecimentos em sala de aula.

Todo o processo, de inserir o "ERP na sala de aula", iniciou através da parceria junto a SAP, pois já era a solução adotada na gestão da universidade. Porém, devido a complexidade, necessidade de especialistas e, principalmente o fato do SAP/R3 ser uma ferramenta complexa e pesada, embora muito poderosa, o grupo de pesquisa buscou outras alternativas de parceria para atuar na área do ensino da teoria de sistemas ERP.
Diversas soluções de mercado, principalmente aquelas voltadas para as médias empresas, foram analisadas. Após muito trabalho, o grupo de pesquisa e a universidade fecharam uma parceria com a empresa Microsiga, empresa brasileira especializada em software de gestão, para o repasse da tecnologia ERP em sala de aula.
Assim com as atividades do grupo, os cursos das áreas de negócio (administração, contabilidade, economia) e os de tecnologia (computação, sistemas de informação e engenharias) serão beneficiados com uso da ferramenta ERP como recurso de apoio no processo de ensino-aprendizagem, oferecendo a oportunidade de conhecimentos sobre a ferramenta, contextualizando os conteúdos de diversas disciplinas, integrando as disciplinas dos respectivos cursos, além de possibilitar a integração entre os diferentes cursos e campi, através de simulações, estudos de caso, e projetos conjuntos de atuação nas empresas da região.

5. MODELO PROPOSTO

A partir da configuração das unidades de ensino da Unisul, é proposto um modelo que permita utilizar a ferramenta ERP como recurso de ensino-aprendizagem, através de uma simulação empresarial envolvendo os seguintes elementos: dois fornecedoras de serviços de suporte e desenvolvimento; empresa fornecedora de matérias-primas; empresa produtora; dois varejistas. A Figura 3 ilustra o modelo abrangente que possibilita a inclusão da ferramenta ERP na Unisul.

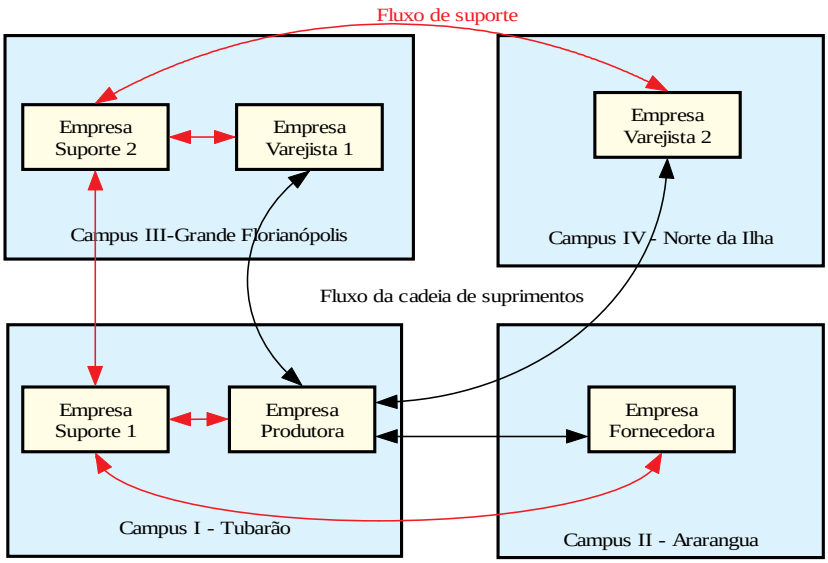


Figura 3 - Modelo abrangente que inclui o sistema ERP na Unisul

Os cenários para o processo de ensino-aprendizagem utilizando o ERP são listados a seguir:

- *nas disciplinas curriculares*: uso do ERP para navegação, realização de transações e atividades no laboratório e exemplificação do conteúdo teórico;
- *integração das disciplinas nos cursos*: uso do ERP através da realização de processos integrados envolvendo várias funcionalidades, bem como resolução de estudos de caso, desenvolvimento de ferramentas de customização, administração da base de dados, configuração do sistema; e
- *integração intercursos e intercampus*: uso do ERP para a simulação de um ambiente empresarial, envolvendo uma cadeia de suprimentos com fornecedor, produtor e varejista, além das empresas de suporte e desenvolvimento da ferramenta, com os alunos desenvolvendo atividades de acordo com sua área de formação.

Os alunos dos cursos de negócio (administração, contabilidade, economia) dos quatro campi são agrupados em quatro empresas representadas na cadeia. Nestas empresas, eles desenvolvem atividades empresarias relacionadas à compras, finanças, produção, vendas e recursos humanos, além das interações comerciais do tipo B2B e de SCM *(suply chain mangement)*.
Já os alunos das áreas tecnológicas (computação, sistemas de informação e engenharias) são reunidos em dois grupos constituindo duas empresas de suporte às quatro empresas da cadeia de suprimentos, onde desenvolverão atividades de desenvolvimento, administração dos sistemas e de suas base de dados, desenvolvimento de customizações, etc.
A estratégia de alocação das empresas fictícias em função dos campi atende aos interesses da Unisul em contextualizar seus conteúdos com a vocação da região onde o respectivo campus está instalado. O critério relacionado aos cursos oferecidos nos campi também é levado em conta para a localização das empresas da seguinte forma:

- empresa fornecedora (instalada no laboratório do Campus Araranguá) é responsável pelas atividades de beneficiamento de matérias-primas;
- empresa produtora (instalada no laboratório Campus Tubarão) é a responsável pelas atividades de compra e montagem de produto;
- empresa varejista 1 (instalada no laboratório do Campus da Grande Florianópolis) e empresa varejista 2 (instalada no laboratório do Campus do Norte da Ilha) responsabilizam-se pela compra e venda de produtos ao consumidor final;
- empresa fornecedora de serviços de suporte 1 (instalada no laboratório do Campus Tubarão) atua no desenvolvimento de aplicativos, customizações e administração da base de dados e dos sistemas das empresa fornecedora (Campus Araranguá) e da produtora (Campus Tubarão);
- empresa fornecedora de serviços de suporte 1 (instalada no laboratório do Campus da Grande Florianópolis) atua no desenvolvimento de aplicativos, customizações e administração da base de dados e dos sistemas das empresas varejistas.

O fluxo nesta cadeia inicia com a elaboração e comercialização de matérias-primas pela empresa denominada fornecedora. A empresa produtora adquire os insumos da empresa anterior e produz um produto que será comercializado pelas empresas varejistas. Operações intraorganizacionais (administrativo-financeiras, produtivas e comerciais) como interorganizacionais (B2B e SCM) são simuladas pelos alunos dos cursos de negócios.
As empresas envolvidas no fluxo de materiais (Fornecedora, Produtora e Varejista) propiciam aos estudantes dos cursos de negócio o desenvolvimento de atividades relacionadas a configuração de cada empresa no sistema, através de atividades de parametrização, bem como do uso das funcionalidades do ERP para registrar, modificar e analisar as operações tanto através da realização de transações nos módulos como na simulação de processos integrados, envolvendo múltiplas funcionalidades além de processos interorganizacionais através de interações de comércio eletrônico.
Já as atividades de cunho tecnológico, desenvolvidas pelos alunos dos cursos de computação, sistemas de informação e engenharia, nas duas empresas de suporte simulam as atividades relativas à da empresa fornecedora do ERP no suporte, auditoria e desenvolvimento do sistema , como as atividades de administração do sistema e da base de dados das quatro empresas da cadeia de suprimentos. A tabela 1 descreve os elementos envolvidos no modelo proposto.
Os alunos das áreas de tecnologia poderão trabalhar nas empresas denominadas de Suporte, num ambiente de desenvolvimento integrado, definindo necessidades e desenvolvendo soluções usando ferramentas de desenvolvimento, de implementação de ERP, customizando um processo ou toda a organização, como também a avaliação dos guias de usuários desde o início da implementação até as fases finais do processo. Também estarão realizando atividades típicas como administrador do sistema das empresas do fluxo de materiais, monitorando e produzindo relatórios para os usuários das empresas do fluxo de materiais.

Tabela 1 – Descrição dos elementos do modelo

Campus	Empresa	Cursos	Atividades	Conectividade
I – Tubarão	Suporte 1	Computação Sistemas de Informação Engenharias	administração da base de dados; desenvolvimento; administração do sistema; consultoria; auditoria dos sistemas.	Empresa produtora (campus I) Empresa fornecedora (campus II) Empresa de suporte (Campus III)
III-Grande Florianópolis	Suporte 2	Computação Sistemas de Informação Engenharias	administração da base de dados; desenvolvimento; administração do sistema; consultoria; auditoria e configuração do sistema.	Empresa varejista 1 (campus III) Empresa varejista 2 (campus IV) Empresa de suporte (Campus I)
II - Araranguá	Fornecedor	Administração Contabilidade	transações e processos; parametrização; processos B2B.	Empresa produtora (campus I) Empresa de suporte (Campus I)
I - Tubarão	Produtor	Administração Contabilidade Economia	Transações e processos; parametrização; processos B2B;	Empresa fornecedora (campus II) Empresas varejista 1 (campus III) Empresa varejista 2(Campus IV)
III-Grande Florianópolis	Varejista 1	Administração Contabilidade	transações e processos; parametrização; processos B2B.	Empresa produtora (campus I) Empresa de suporte (Campus III)
IV Norte da Ilha	Varejista 2	Administração Contabilidade	transações e processos; parametrização; processos B2B.	Empresa produtora (campus I) Empresa de suporte (Campus III)

6. CONCLUSÕES

Este trabalho apresentou uma proposta de modelo abrangente de utilização de um sistema ERP como recurso de apoio ao processo de ensino-aprendizagem a ser implementado nos cursos tecnológicos e de negócios nos campi da Unisul.

A possibilidade de utilização de uma ferramenta ERP como recurso de apoio ao processo de ensino- aprendizagem é muito valiosa. Permitindo, através de uma simulação, a configuração de um sistema que represente diferentes tipos de agentes -fornecedor, produtor, varejistas e empresas de suporte tecnológico – é possível realizar inúmeras atividades de cunho didático. Conhecimentos relacionados às atividades e processos empresariais, incluindo atividades de B2B e SCM, bem como conteúdos tecnológicos (linguagem de programação, administração do sistema, desenvolvimento, dentre outras) geram uma gama bastante ampla de oportunidades à alunos e professores no desenvolvimento de habilidades e competências exigidos pelo mercado.

A Unisul, ao implementar um sistema ERP em suas atividades administrativas e criar um grupo de pesquisas em sistemas ERP, está delineando novas oportunidades ao mundo empresarial, formando profissionais mais capacitados para as demandas do mercado, como ao meio científico, ao criar um substrato para o desenvolvimento de novas técnicas, modelos, metodologias e pesquisas nas áreas de TI e de negócios.

Expandir este modelo através da inclusão de outras universidades, permitirá a ampliação das interações comerciais e tecnológicas na cadeia proposta, com empresas relacionadas a vocação da instituição e sua região, além de estimular a colaboração científica entre professores, pesquisadores e alunos participantes.

Referências Bibliográficas

[1] Albertão, S. E. *ERP: sistemas de gestão empresarial: metodologia para avaliação, seleção e implantação*. São Paulo: Iglu, 2001.

[2] Antonucci, Y.L. and Zur Meuhlen, M. Deployment of Business to Business Scenarios in ERP Education: Evaluation and Experiences from an International Collaboration, In: PROCEEDINGS OF THE SEVENTH AMERICAS CONFERENCE ON INFORMATION SYSTEMS, Miami – USA. Anais… Miami: VII ACOIS, 2001, pp. 998-1004.

[3] Dávalos, R. V. e López, O. C. Uma abordagem da implantação de um ERP visando apoio às atividades administrativas e de ensino. In: 3ª CONFERÊNCIA DA ASSOCIAÇÃO PORTUGUESA DE SISTEMAS DE INFORMAÇÃO, Coimbra - Portugal. *Anais*... Coimbra: CAPSI, 2002.

[4] Dávalos, R. V. e Platt, A. A. Implantação de um Sistema Integrado de Gestão visando apoio às atividades universitárias. In: XXVII CONGRESSO BRASILEIRO DE ENSINO E ENGENHARIA – COBENGE 2002, Piracicaba – SP. *Anais*... Piracicaba: COBENGE, 2002.

[5] Davenport, T. H. Putting the enterprise into the enterprise system. *Harvard Business Review*. Julho-Agosto, 1998, p.121-131.

[6] Davenport, T. H. Teaching about Reengineering. Association for Information Systems – American Conference on Information Systems. Disponível em:< http://hsb.baylor.edu/ramsower/acis/papers/ davenpor.htm > Acesso em : 22/08/2002.

[7] Guthrie, R W. and Guthrie, R.A.. "Integration of Enterprise System Software in the Undergraduate Curriculum," Proceedings of ISECON 2000, Philadelphia, Vol. 17, No. 301. Hall, R. "Enterprise Resource Planning Systems Planning Systems and Organizational Change: Transforming Work Organizations?" Strategic Change, Vol. 11, 2002, pp. 263-270.

[8] Haberkorn, E. *Gestão Empresarial com ERP*. São Paulo, Microsiga Software SA, 2003, 674p.

[9] Hammer, M. *Além da reengenharia*. Rio de Janeiro: Campus, 1997.

[10] Hawking, P. and McCarthy, B. The ERP eLearning Model for the Delivery of ERP (SAP R/3) Curriculum into the Asian Region. Informing Science; Challenges to Informing Clients: A Transdisciplinary Approach; June 2001.

[11] Hawking, P., Foster ,S., Bassett, P. An Applied Approach to Teaching HR Concepts Using an ERP System. Informing Science InSITE - "Where Parallels Intersect" June 2002. Pierson, M.M. and Pierson, B.L. Beginnings and Endings: Keys to Better Engineering Technical Writing. *IEEE Trans. On Professional Communication*. Vol 40, No. 4, (December, 1997), pp. 299-304.

[12] Mendes, J. V. e Escrivão Filho, E. Sistemas integrados de gestão ERP em pequenas empresas: m confronto entre o referencial teórico e a prática empresarial. Gestão & Produção, v. 9, n.3, dez.2002, p.277-296.

[13] Watson, E. E. and Schneider, H. Using Erp Systems In Education. Communications of AIS (Association of Information System) Volume 1, Article 9; February 1999.

[14] Wieder, B. ERP – Software integration at Australian Universities – recent developments in integrated business education. Proceedings of CTI Accounting Finance & Management Conference. Grã-Bretanha, 1999.

Cuatro Universidades y Un Doctorado, o Colaboración vs. Competencia en Educación Superior

Francisco J. Torres-Rojas, Rodrigo Bogarín, César Garita
{torres, rbogarin, cesar}@ic-itcr.ac.cr
Instituto Tecnológico de Costa Rica y
Centro de Investigación en Computación e Informática Avanzada (*CIenCIA*)

Gabriela Marín Raventós, Vladimir Lara
{gmarin, vlara}@cariari.ucr.ac.cr
Universidad de Costa Rica y
Centro de Investigación en Computación e Informática Avanzada (*CIenCIA*)

Costa Rica

Resumen. Después de décadas de competencia, las Escuelas de Computación de las cuatro universidades públicas de Costa Rica proponen un programa conjunto de Doctorado en Ciencias de la Computación, el cual se encuentra en las etapas finales de aprobación por parte de las autoridades correspondientes. Este artículo es un breve reporte del proceso de creación, de las soluciones que encontramos a problemas académicos y administrativos, y de la propuesta actual.

Palabras Clave: Programa de Doctorado, Educación Superior, Postgrado, Costa Rica.

"La République n'a pas besoin de savants"
Jean-Paul Marat, *al condenar a morir en la guillotina a Lavoisier, considerado el padre de la química moderna*

1 Introducción

En años recientes, Costa Rica ha decidido dejar de ser un simple consumidor de Ciencia y Tecnología extranjera, y quiere hacer lo necesario para constituirse en un actor a ser tomado en cuenta a nivel mundial en estos campos. Creemos firmemente que la Ciencia no es una labor egoísta o meramente académica: sin la Ciencia, el progreso y, muy posiblemente, la supervivencia misma de los pueblos no pueden ser garantizados. Aunque la investigación científica tiende a ser cara, sería ingenuo de parte de los países con menos recursos rehuir a su participación en esta actividad, ya que de otra forma el ciclo de dependencia tecnológica nunca podrá romperse. En particular, las Ciencias de la Computación y la Informática se han constituido en uno de los referentes más notables de las capacidades científicas y tecnológicas de nuestro país.

En Costa Rica hay cuatro universidades públicas, a saber: *Universidad de Costa Rica* (**UCR**), *Instituto Tecnológico de Costa Rica* (**ITCR**), *Universidad Nacional* (**UNA**) y *Universidad Estatal a Distancia* (**UNED**). En las tres primeras hay programas muy establecidos que forman profesionales en Computación e Informática desde hace más de 25 años. Con pequeñas diferencias de enfoque, se tienen programas de Maestría en Computación en el **ITCR** (desde hace más de 17 años), **UCR** y **UNA**. La modalidad de maestría profesional en estas universidades tiene mucho más demanda que la maestría académica. A pesar de esto, la **UCR** y el **ITCR** continuamente tratan de incentivar los programas de maestría académica para apoyar la investigación.

Tradicionalmente, estos programas han competido por atraer los mejores estudiantes y por graduar los mejores profesionales, manteniendo casi siempre una "elegante" rivalidad entre las escuelas correspondientes. La competencia ha sido dura y durante muchos años, especialmente entre el **ITCR** y la **UCR** (consideradas las universidades con programas de mayor calidad). Producto de esta competencia, la colaboración entre estos programas ha sido casi nula. Sin embargo, debemos decir que esta competencia ha traído como beneficio una mejora casi constante en la calidad y alcances de cada programa. Los egresados en computación de estas universidades cuentan con un muy merecido prestigio técnico de nivel internacional, y constituyen las contrataciones preferenciales en las instituciones y empresas nacionales.

Al mismo tiempo, o tal vez como resultado de lo anterior, la industria nacional de software ha experimentado un enorme crecimiento cuantitativo y cualitativo en la última decada. Muchas empresas son exportadoras de software a mercados internacionales altamente competitivos, inclusive varias son reconocidas como líderes mundiales en sus campos de especialización. Dicha industria ha llegado a un punto de madurez donde las inversiones en Investigación y Desarrollo son no sólo normales, sino que casi indispensables. Ya que muchas de estas empresas fueron fundadas por profesores y egresados de estas universidades, se tiene una saludable, aunque todavía pequeña, cooperación entre industria y academia en esta área.

Aprovechando la situación descrita previamente, el *Consejo Nacional de Rectores* (CONARE), ente que reúne a los rectores de las cuatro universidades públicas de Costa Rica, dio el mandato para la definición y puesta en marcha de un ambicioso **Programa de Doctorado en Ciencias de la Computación** hecho en forma conjunta por las cuatro universidades. En forma independiente, ninguna universidad tiene con holgura la masa crítica de profesores con doctorado en Computación necesaria para crear un Programa de Doctorado. Sin embargo, combinando el personal y los recursos de estas universidades sí es factible tal intento.

Así, se convocó a una comisión interuniversitaria proveniente de las escuelas de Computación de las universidades de CONARE para diseñar este programa doctoral. Se seleccionó al *Centro Nacional de Alta Tecnología* (CENAT) para que funcionara como catalizador de este proyecto, proporcionando un terreno neutral para la discusión y análisis. Dado lo valioso del objetivo común, se reemplazó con cooperación y colaboración a la competencia de décadas. Se establecieron 3 directrices en el trabajo de la comisión:

- *Cooperación interuniversitaria*
- *Investigación científica como prioridad*
- *Calidad de nivel internacional.*

Esta comisión ha tenido que hacer un esfuerzo importante, a veces titánico, para considerar las regulaciones de cada universidad y los reglamentos de CONARE relativos a estudios de postgrado. Se consultó una amplia bibliografía respecto a investigación científica y programas doctorales [1, 2, 3, 4, 5, 6, 10, 11, 12, 13, 15] y se buscaron las sugerencias, comentarios y retroalimentación de expertos nacionales e internacionales (Alemania, Chile, Cuba, España, Estados Unidos, Francia e Italia). Versiones previas de la propuesta, ahora actualizadas, fueron presentadas en [9, 14]. Sin embargo, los aportes más significativos vinieron de las experiencias particulares de los miembros de la comisión, quienes realizaron sus estudios doctorales en diversas universidades europeas y estadounidenses. La influencia inescapable de estos centros de estudio debe ser evidente en este artículo y en la amplia documentación que ha sido generada.

Se identifican dos elementos claves:

- *Creación de un Centro de Investigación avanzada en la disciplina, que reunirá en nuestro país a un grupo importante de doctores en Ciencias de la Computación dedicados a la investigación científica, la cual será cuantificada y evaluada estrictamente de acuerdo al número de publicaciones que este grupo produzca.*
- *Establecimiento de un programa académico interuniversitario de Doctorado en Ciencias de la Computación, entendido como el subproducto más importante del Centro de Investigación mencionado anteriormente.*

Al momento de escribir este artículo, el proyecto se encuentra en las fases finales de aprobación por parte de las autoridades correspondientes. El *Centro de Investigación en Computación e Informática Avanzada* es descrito en la Sección **2**. La estructura administrativa sugerida para el Doctorado se encuentra en la Sección **3**. La Sección **4** detalla los requisitos de ingreso a este Programa. Las diversas piedras millares que el estudiante debe alcanzar como parte de sus estudios doctorales son presentadas en la Sección **5**. El perfil profesional y académico del graduado del programa propuesto es brevemente analizado en la Sección **6**. Finalmente, el artículo concluye en la Sección **7**.

2 Centro de Investigación en Computación e Informática Avanzada (*CIenCIA*)

En su famosa conferencia "*The Uncertainty of Science*" [7], Richard Feynman explora tres pilares fundamentales de lo que entendemos por Ciencia. En primer lugar, la Ciencia moderna se caracteriza por su riguroso método o procedimiento para obtener conocimiento nuevo, con hipótesis y teorías delimitadas por la inexorable navaja de Occam, experimentos objetivos y reproducibles, y la comunicación apropiada de los resultados obtenidos para someterse al dictamen exigente de otros científicos [4, 6], siguiendo con Feynman, podemos decir que los científicos se dedican al "placer de hallar cosas" [8]. En segundo lugar, Feynman se concentra en el cúmulo del conocimiento publicado de una disciplina. Así, la colección de todos los "*papers*", tesis, libros y conferencias publicados por todos

los investigadores del mundo constituyen el acervo científico de la humanidad, y, consecuentemente, trabajos nunca publicados o investigaciones guardadas en gavetas no son parte de la Ciencia [4, 15]. Finalmente, la sociedad está particularmente interesada en la "tecnología" o aplicación práctica del conocimiento obtenido por los científicos, para retribuir en la forma de soluciones prácticas la inversión que se ha hecho en Ciencia [6, 7, 8]. En resumen, notamos tres elementos claves en el trabajo científico: método sistemático, riguroso y objetivo; publicación de resultados; y aplicación del conocimiento obtenido.

Un "*Centro de Investigación Científica*" es una entidad que reúne investigadores con el objetivo fundamental de producir y divulgar nuevo conocimiento científico. Este esfuerzo procura en una forma u otra el cumplimiento de los tres elementos característicos de la Ciencia mencionados arriba. El **Centro de Investigación en Computación e Informática Avanzada** (*CIenCIA*), es la respuesta que la comunidad científica nacional en Computación da a este reto. En la medida de lo posible, los investigadores asociados al *CIenCIA* serán dotados con facilidades y recursos que les permitan completar su trabajo científico en Computación e Informática, someter sus resultados al dictamen de sus colegas internacionales, publicar y divulgar su investigación, y buscar la aplicación práctica de sus hallazgos compartiendo resultados con la industria nacional.

Aparte de la investigación misma, uno de los subproductos más importantes del *CIenCIA* será la formación de un programa de Doctorado en Ciencias de la Computación, adonde estudiantes de postgrado, altamente calificados y seleccionados cuidadosamente, recibirán una formación rigurosa y comprensiva en esta disciplina, colaborarán en los diversos proyectos de investigación de sus profesores, y completarán sus propias investigaciones originales en la forma de disertaciones doctorales. Toda esta investigación será canalizada a través del *CIenCIA*.

La investigación sin publicación es estéril, autoreferente, intrascendente y de calidad dudosa [4, 15]. Por lo tanto, el objetivo fundamental del *CIenCIA* es producir y, sobre todo, **publicar** investigación científica del más alto nivel en Computación e Informática. Se plantean como objetivos específicos: identificar y reunir a los mejores investigadores del país y la región en Computación e Informática; crear un ambiente agradable y estimulante, dotado de los mejores recursos y facilidades posibles para la producción científica en Computación e Informática, y ponerlo en forma expedita al servicio de investigadores nacionales e internacionales asociados a *CIenCIA*; establecer fuertes contactos con el sector académico nacional, identificado principalmente en el Programa de Doctorado en Ciencias de la Computación, y con el sector industrial nacional en el área de Computación; colaborar, siempre con la investigación científica como prioridad, en la formación de recurso humano especialista en la disciplina; y exigir a sus investigadores asociados la publicación de resultados ante la comunidad científica internacional.

Los proyectos de investigación del *CIenCIA* pueden ser intrínsecamente de Computación e Informática (por ejemplo, Ingeniería de Software, Bases de Datos, Redes de Computadores, Sistemas Operativos, Lenguajes de Programación, etc.) o ser de naturaleza interdisciplinaria con un componente significativo de interés para la comunidad científica en Computación e Informática (por ejemplo, Biología Molecular Computacional, Simulación, Investigación de Operaciones, Matemáticas, etc.). A su vez, pueden ser de tipo estrictamente teórico o con fuertes implicaciones prácticas y conexiones con la industria nacional. Debido a la fuerte relación con el Programa de Doctorado en Computación, se espera que muchos de estos proyectos sean parte de las investigaciones doctorales de los estudiantes de este programa, aunque tampoco se descartan como generadores de investigación publicable las tesis de Maestría.

Uno de los principales parámetros (y más fácilmente cuantificable) para evaluar la efectividad de la labor del *CIenCIA* es el número de publicaciones realizadas a su nombre por sus asociados. Estas publicaciones buscarán difundir los resultados de nuestras investigaciones y, sobre todo, recibir retroalimentación respecto a ellas. Así, la labor del *CIenCIA* puede ser evaluada en tres aspectos: **Cantidad,** número de Reportes Técnicos producidos por semestre; **Calidad,** número de publicaciones en Conferencias y *journals* de prestigio mundial; e **Influencia,** número de referencias en otras investigaciones de las publicaciones auspiciadas por el *CIenCIA*. Por otro lado, y sin perder de la mira la investigación científica y su publicación como objetivo fundamental de *CIenCIA*, hay una diversidad de actividades colaterales que el Centro podría desarrollar:

- "*Broker*" de proyectos de investigación: *CIenCIA* recopilará y anunciará problemas de industrias nacionales, así como "Expresiones de Interés" de los académicos.
- Canalizar posibles fuentes de financiamiento para llevar a cabo investigación.
- Organizar actividades que fomenten el intercambio de ideas (por ejemplo, Conferencias Científicas en Computación, nacionales e internacionales), la vinculación industria-universidad, y la internacionalización de los resultados (charlas, visitas a universidades y colegios, etc.).
- Crear "Redes de Excelencia" a travez de las cuales expertos de ciertos temas pueden reunirse (física o virtualmente) e intercambiar ideas.
- Publicación periódica de una revista.

3 Estructura Administrativa del Programa de Doctorado

Se considerará que un profesor es "*miembro*" del Programa de Doctorado si está afiliado a alguna de las escuelas de Computación de alguna universidad de CONARE, con una dedicación mayor a medio tiempo, tiene un grado mínimo de doctor, y es profesor e investigador activo (con un número adecuado de publicaciones recientes) dentro del Programa de Doctorado. Todos los profesores miembros tienen el **derecho** de ser parte de la Asamblea Académica del Doctorado con voz y voto. Esta Asamblea es la máxima autoridad del Programa de Doctorado y se reunirá por lo menos dos veces por semestre, deberá escoger entre sus miembros a un **Director del Doctorado** nombrado por períodos de 4 años. Las funciones del Director incluyen coordinar el funcionamiento del Programa y vigilar el cumplimiento de los objetivos del mismo. Habrá un **Comité Ejecutivo del Doctorado** (*C.E.D.*), formado por un representante nombrado por cada una de las universidades públicas (con un grado mínimo de doctor, preferiblemente en Ciencias de la Computación), el Director del Doctorado, dos miembros de la Asamblea Académica, y un representante del CENAT. El *C.E.D.* se reunirá periódicamente, convocados por el Director del Doctorado. Su principal función es implementar las directrices establecidas por la Asamblea Académica y tomar todas las decisiones administrativas necesarias para el funcionamiento normal del Programa (por ejemplo, Admisión de Estudiantes, reconocimiento de cursos, aprobación de Comités de Tesis, etc.).

4 Ingreso al Doctorado

En cualquier disciplina, un programa de doctorado requiere que sus estudiantes tengan una dedicación de tiempo completo durante los años que toma su educación y entrenamiento. Es muy importante que el programa tenga claras políticas de ingreso que maximicen la probabilidad de éxito de los estudiantes. Obviamente, esto beneficia al estudiante, al programa, a sus profesores y a las investigaciones producidas. El *C.E.D.*, en consulta con todos los profesores e investigadores del programa, seleccionará los estudiantes a ser admitidos en el programa tomando en cuenta su capacidad intelectual, educación previa, características personales, cartas de recomendación y potencial para investigación de los candidatos. En la gran mayoría de los casos, se *recomienda* que los candidatos a ingresar al Programa de Doctorado cuenten con una Maestría previa en Ciencias de la Computación, sin embargo, en casos excepcionales, el *C.E.D.* podría admitir estudiantes con un Bachillerato, o con una Maestría en otra disciplina. Los interesados deben presentar:

- ***Formularios*** *de solicitud de ingreso al programa completos.*
- ***Ensayo*** *adonde el interesado describa sus metas profesionales, sus intereses de investigación, sus expectativas respecto al programa y sus planes futuros como científico o académico. El C.E.D. evaluará características tales como madurez, motivación, creatividad, habilidad de comunicación, y seriedad de sus intenciones.*
- ***Curriculum Vitae*** *actualizado.*
- ***Documentación*** *original y fotocopias de todos sus estudios universitarios previos. Incluir diplomas y certificados de notas. El C.E.D. establecerá un promedio mínimo de notas.*
- ***Resultados del*** *Graduate Record Examination (**GRE**) en Ciencias de la Computación. El propósito de este requisito es verificar que el candidato tiene el conocimiento equivalente a un Bachillerato en Ciencias de la Computación. El C.E.D. establecerá un puntaje mínimo aceptable.*
- ***Resultados del*** *Test of English as a Foreign Language (**TOEFL**).*
- ***Cartas de Recomendación****, preferiblemente de personas con título de Doctorado en Ciencias de la Computación que sepan de las cualidades académicas del interesado y que puedan juzgar las posibilidades reales del mismo de tener un buen rendimiento como investigador en un programa de doctorado.*
- *Cualquier otra información (publicaciones, patentes, premios obtenidos, certificaciones de habilidades especiales, etc.) que el interesado considere pertinente.*

5 Estructura General

La amplísima disciplina que enmarcamos como "*Ciencias de la Computación*" es, con sus poco más de 50 años de existir, definitivamente un campo muy joven cuando lo comparamos con otras áreas del conocimiento humano que se han desarrollado por siglos e inclusive milenios. Para muchos expertos en la materia, es evidente que eventualmente la Computación se separará en campos de estudio complementarios pero con características propias, como se nota actualmente, por ejemplo, en las diversas ramas de la Ingeniería. Sin embargo, en este instante nos encontramos en un punto previo a esta escisión, y aún no está del todo claro cual será el cuerpo de conocimientos que definirá cada área independiente. Además, se avanza a gran rapidez, haciendo obsoleto el conocimiento establecido o adquirido hace pocos años.

Dada esta situación, un Doctor en Ciencias de la Computación requiere tener bases sólidas y lo más actualizadas posibles en una serie de áreas reconocidas como fundamentales en el campo, y con una profundidad mucho mayor que la que usualmente se tiene en los programa de Maestría convencionales. Es por tal motivo que la comisión interuniversitaria concluyó que era indispensable contar con una serie de Seminarios Doctorales y cursos de Especialidad obligatorios (ver subsecciones **5.1**, **5.2** y **5.3**) para garantizar una formación coherente y actualizada a los futuros graduados del Programa de Doctorado. Esta práctica es corriente en muchos programas de doctorado de prestigio mundial. Nótese, además, que la presencia de los cursos da un punto adicional de control de calidad y que es consistente con un programa de carácter regional donde se recibirán estudiantes con cierta variedad en su formación previa.

Por supuesto, la esencia misma de un doctorado es la investigación científica y sería absurdo siquiera proponer un programa de doctorado integrado únicamente por cursos académicos ordinarios. En esta propuesta, se busca un balance entre cursos (los cuales serán muy orientados a la investigación científica) y la investigación original de cada estudiante. Los cursos de *Investigación Dirigida* constituyen la mayoría porcentual de los créditos asignados al programa de estudios y satisfacen este componente de investigación.

El programa de Doctorado en Ciencias de la Computación prepara individuos excepcionalmente calificados para que se conviertan en profesionales especialistas en esta disciplina. El grado de "doctor" (*Ph.D.*) se confiere por el desarrollo de investigación original que resulte en una contribución significativa al cúmulo de conocimiento del área. El programa tiene varias fases y requiere, partiendo de una Maestría en Ciencias de la Computación, de un mínimo de 6 semestres a tiempo completo. Con cierta regularidad (por lo menos una vez al año) el rendimiento de todos los estudiantes será evaluado para detectar posibles deficiencias o resultados no satisfactorios. Cada estudiante contará con un **profesor asesor** a lo largo de sus estudios de doctorado. Es responsabilidad de cada estudiante seleccionar a su asesor lo antes posible (inicialmente las funciones de asesoría serán asumidas por el Director del Doctorado). Este profesor debe ser *miembro* del Doctorado y debe trabajar en el área en la que el estudiante desea realizar su investigación.

La Figura **1** muestra un diagrama del Plan de Cursos recomendado para estudiantes admitidos al programa con una Maestría previa en Ciencias de la Computación (por motivos de espacio, se omite en este artículo la malla curricular para estudiantes admitidos excepcionalmente con un bachillerato en Computación). Las siguientes subsecciones describen las principales "piedras millares" que el estudiante debe alcanzar en el Programa Propuesto. .

5.1 Núcleo Académico

Como mecanismo de control de calidad y para garantizar una base común de conocimientos, se establecen cuatro *Seminarios Doctorales* obligatorios de 4 créditos cada uno. Estos seminarios no podrán ser reemplazados por otros cursos del currículum ni por cursos tomados previamente. Los estudiantes deben cumplir con este requisito tan pronto como sea posible. Los Seminarios Doctorales son:

- *Introducción a Estudios de Doctorado.*
- *Diseño de Experimentos.*
- *Desarrollo de Software.*
- *Sistemas Computacionales.*

5.2 Componente de Especialidad

El profesor asesor establecerá que cursos, por lo menos 2, debe tomar el estudiante dentro de su área específica de especialización e investigación. Es factible que, a solicitud expresa del estudiante y luego de presentar documentación exhaustiva al respecto, el *C.E.D.* reconozca algunos cursos tomados previamente (e.g., como parte de una Maestría previa en Ciencias de la Computación) para cumplir con parte de este requisito, siempre y cuando se cuente con la aprobación del profesor asesor. Cursos que no hayan sido impartidos por profesores con doctorados no podrán ser considerados. Cada curso de especialidad aprobado dará 4 créditos.

5.3 Conocimiento Básico - Exámenes Comprensivos

El programa establece 5 áreas indispensables que todo estudiante de doctorado en Ciencias de la Computación debe dominar:

- *Análisis de Algoritmos.*
- *Sistemas Operativos.*
- *Diseño de Lenguajes de Programación.*
- *Bases de Datos.*

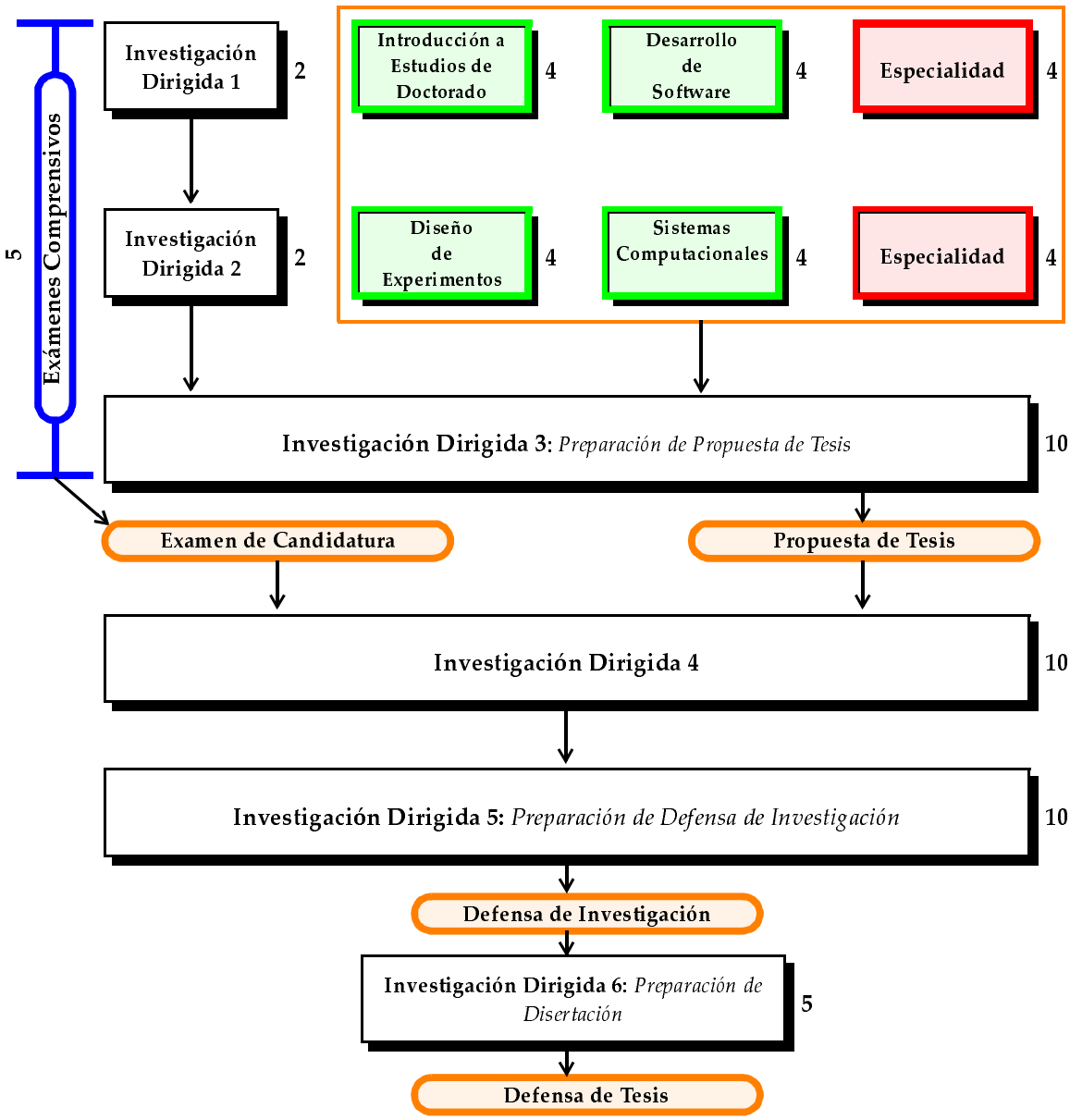


Figura 1. *Plan de Cursos para Estudiantes con Maestría en Ciencias de la Computación*

- *Redes de Computadores.*

Estos temas serán evaluados con exámenes comprensivos individuales para cada una de las áreas definidas (se asigna 1 crédito a cada examen). Nótese que este no es un requisito de ingreso, sino un requisito de permanencia. Los estudiantes deben aprobar todos los exámenes durante sus primeros tres semestres en el programa de doctorado. Si el rendimiento mostrado en algunos de estos exámenes no es satisfactorio, el estudiante tendrá oportunidad de tomar, por una única vez más, los exámenes requeridos. Cada examen comprensivo tiene el valor de un crédito académico. Se entiende que el conocimiento requerido en estas áreas es correspondiente a cursos de nivel de una Maestría en Ciencias de la Computación. El programa de doctorado como tal no ofrecerá cursos para satisfacer este requisito, sino que dará a conocer los temas a ser evaluados en cada uno de los exámenes comprensivos. La aprobación de los exámenes comprensivos es obligatoria para todo estudiante que desee continuar en el programa de doctorado, y ningún estudiante podrá ser eximido de tomarlos y aprobarlos, bajo ninguna circunstancia.

5.4 Créditos de Investigación

Desde el primer semestre en el programa, y durante todo el tiempo que permanezcan en el programa de doctorado, todos los estudiantes deben matricular **todos** los semestres por lo menos un curso de *Investigación Dirigida*, bajo la tutela de alguno de los investigadores del programa, el cual orientará al estudiante en sus lecturas, lo hará participar en sus proyectos de investigación y lo evaluará de la manera que considere apropiada (e.g., preparando un *paper* al final del semestre). Dicho curso recibirá una nota al final de cada semestre. Como puede notarse en el Plan de estudios mostrado en la Figura **1**, estos cursos se identifican como *Investigación Dirigida 1*, hasta *Investigación Dirigida 6*.

Estos cursos estarán totalmente orientados a la investigación. Siempre que sea posible se tendrá como meta la publicación (conjunta o individual) de artículos en conferencias internacionales o revistas de prestigio, o que el estudiante se prepare para completar exitosamente la siguiente fase específica del plan de estudios. El tiempo dedicado a estos cursos será utilizado en la investigación particular de cada estudiante y a colaborar con los proyectos de investigación en que esté involucrado. Así, la preparación de la Propuesta de Tesis, la Defensa de Investigación y la Defensa de Tesis, serán hechas bajo la dirección del profesor asesor en las horas asignadas a Investigación Dirigida.

Para el Programa de Doctorado es fundamental la exposición internacional de sus estudiantes y sus trabajos. Esta exposición puede ser lograda ya sea mediante la publicación y presentación de las investigaciones científicas del estudiante en múltiples conferencias internacionales de prestigio, o con una combinación de presentaciones en conferencias y una pasantía en el extranjero, no mayor a un semestre, realizada en alguna Universidad o Centro de Investigación de calidad mundial. Dicha pasantía cubriría el requisito de "investigación dirigida" de un semestre y debe ser realizada antes de la *Defensa de Investigación*.

5.5 Examen de Candidatura

Cuando un estudiante haya avanzado lo suficiente en sus estudios de doctorado (aproximadamente después de 4 o 5 semestres), un área específica de investigación debe ser seleccionada. El estudiante debe demostrar un conocimiento profundo en esta área, aprobando el respectivo Examen de Candidatura para dicha área. Este examen es un paso fundamental para convertirse en candidato a doctor. El estudiante debe identificar un asesor académico (normalmente, esta función es realizada por el mismo profesor asesor) y por lo menos otros dos profesores para que integren el "Comité del Examen de Candidatura". Este comité se encargará de supervisar todo el proceso de este examen. Cada área de investigación publicará la lista del material que considera que el estudiante debe conocer (e.g., lista de libros, papers, lenguajes de programación, software, sistemas operativos, etc.). El examen consta de tres partes:

- ***Examen Escrito***. *Un examen comprensivo de un día de duración. Queda a criterio de los examinadores si dicho examen es a libro abierto o no. Este componente evalua la profundidad de los conocimientos del estudiante en el área seleccionada.*
- ***Pregunta de Investigación***. *El Comité le propondrá al estudiante un problema o pregunta de investigación la cual debe explorar e (idealmente) resolver en, a lo sumo, una semana de tiempo. El estudiante debe preparar una presentación con su respuesta o escribir un esquema detallado de un posible artículo científico.*
- ***Defensa Oral***. *El estudiante presentará su respuesta a la pregunta de investigación ante el Comité del Examen. El comité interrogará al estudiante respecto a su presentación, la pregunta de investigación, el examen escrito o respecto a los temas que considere pertinentes.*

El Comité del Examen se encargará de dar la evaluación final respecto al examen, considerando los resultados de todos los componentes. Si el rendimiento en el examen de candidatura no fuera satisfactorio, el estudiante podrá repetirlo, por una única vez, en un tiempo fijado por el Comité.

5.6 Propuesta de Tesis

La *Propuesta de Tesis* tiene dos componentes: **Documento Escrito** y **Defensa Oral** de dicho documento. En el **Documento Escrito** el estudiante describe con el mayor detalle posible el proyecto de investigación a ser realizado. El objetivo principal es establecer la importancia y originalidad del trabajo, pero sobre todo su factibilidad. El estudiante debe demostrar dominio del tema particular y de los trabajos más relevantes de otros investigadores en el área. Se debe detallar cuanto de la investigación ya ha sido concluida y publicada por el estudiante, y que elementos están pendientes de ser explorados. Se deben listar los recursos de todo tipo que serán necesarios para completar el proyecto (e.g. hardware y software, bibliografía, equipo humano, colaboración de empresas o instituciones, visitas a centros de investigación, etc.) y un cronograma detallado de actividades. La propuesta tendrá una naturaleza semejante a un "contrato" entre el estudiante y el programa de doctorado. Se debe realizar una **Defensa Oral** y pública de la Propuesta de Tesis ante un comité de por lo menos 3 profesores del programa de doctorado, junto con

cualquier otro público interesado. El comité será seleccionado por el estudiante y su profesor asesor (el cual a su vez es miembro de este comité). Aunque no es indispensable, sería conveniente que dicho comité sea el núcleo del posterior Comité de Tesis. La defensa oral será moderada por el profesor asesor. El comité decidirá si la propuesta del estudiante satisface los requisitos que se esperan de una investigación doctoral y, en cualquier caso, hará recomendaciones al estudiante respecto al proyecto propuesto. Un estudiante que haya aprobado todos los requisitos de cursos, su Examen de Candidatura y cuya Propuesta de Tesis haya sido aprobada será *Candidato a Doctor.*

5.7 Defensa de Investigación[1]

Por lo menos un semestre antes de su eventual graduación, el estudiante deberá defender en forma pública su investigación ante un Comité evaluador. Es responsabilidad del estudiante anunciar ampliamente esta actividad a la comunidad académica del Programa de Doctorado y de los departamentos y escuelas de Informática y Computación de todas las universidades públicas del país, indicando el tema de su investigación, así como la fecha, hora y lugar de la defensa. Al momento de esta defensa, la investigación del estudiante debe estar terminada y se debe contar con documentos escritos que la describan (i.e., borrador de la disertación, artículos publicados, resúmenes de resultados, reportes técnicos, etc.). Sin embargo, no es necesario que la Disertación Doctoral esté terminada para entonces. La investigación debe ser presentada y defendida ante un comité de por lo menos 5 profesores, que incluye a su profesor asesor y a un miembro externo al Programa de Doctorado (puede haber un máximo de 2 miembros externos). Es indispensable que todos los miembros del Comité estén presentes en el momento de la defensa.

Esta actividad debe demostrar que la investigación está completa, y que la misma es una contribución original e importante al cuerpo de conocimientos de las Ciencias de la Computación en general y a la especialidad del estudiante en particular. Se verificará que el trabajo presentado se ajuste razonablemente bien a lo descrito en la Propuesta de Tesis. La defensa será moderada por el profesor asesor. Después de una deliberación en privado, el Comité presentará sus comentarios a la investigación del estudiante, indicando, principalmente, si la investigación es satisfactoria para el Comité o si la encuentran deficiente. En todo caso, el Comité explicará los cambios que el estudiante debe realizar al replantear su investigación o antes de completar su Disertación Doctoral. También el Comité podría recomendar cuales partes o resultados individuales de la investigación sería conveniente publicar.

5.8 Disertación Doctoral

La Disertación Doctoral es el documento final donde el estudiante presenta los resultados de su investigación, la cual fue previamente defendida y aceptada por el Programa de Doctorado. El problema investigado debe ser descrito detalladamente, junto con la metodología seguida, los experimentos realizados, si fuera del caso, y los resultados finales. Además se espera una revisión completa y profunda de otros trabajos de investigación relacionados con el tema y una descripción del trabajo futuro que se puede desprender de los resultados encontrados por el estudiante. Este documento debe satisfacer todos los requerimientos que el Comité de su respectiva Defensa de Investigación recomendó, y debe seguir estrictamente todas las regulaciones de forma que el Programa de Doctorado establezca.

5.9 Defensa de Tesis

La última etapa en los estudios de doctorado es la Defensa de Tesis. El procedimiento será similar a los previamente descritos para la presentación de la Propuesta de Investigación y para la Defensa de Investigación. Su principal propósito es hacer una presentación formal de los resultados de investigación y verificar que todas las observaciones hechas en la Defensa de Investigación hayan sido resueltas satisfactoriamente por el estudiante. Es responsabilidad del estudiante anunciar ampliamente esta actividad a la comunidad académica del Programa de Doctorado y de los departamentos y escuelas de Informática y Computación de todas las universidades públicas del país, indicando el tema de su tesis, así como la fecha, hora y lugar de la defensa.

La tesis debe ser presentada y defendida ante un comité de por lo menos 5 profesores, que incluye a su profesor asesor y a un miembro externo al Programa de Doctorado. Es recomendable que todos los miembros del Comité estén presentes en el momento de la defensa de tesis. La defensa será moderada por el profesor asesor. Después de una deliberación en privado, el Comité presentará sus comentarios a la tesis del estudiante, indicando, principalmente, si la investigación es satisfactoria para el Comité o si la encuentran deficiente. Una vez que la Defensa de Tesis haya sido aprobada exitosamente, el estudiante recibirá el título de *Ph.D.*

1. Muchos elementos de este documento fueron influenciados por diversas universidades. En particular, la idea de la *Defensa de Investigación* fue tomada del programa doctoral de *Georgia Tech.*

5.10 Requisitos Adicionales

Aparte de los componentes mencionados en los párrafos anteriores, se requiere que el estudiante haya publicado por lo menos dos *papers* en conferencias internacionales de prestigio o revistas especializadas adonde su trabajo haya sido sometido al arbitraje anónimo de especialistas en el área ("*peer review*").

5.11 Cursos Adicionales para Estudiantes Admitidos con Bachillerato Universitario

Para satisfacer las normativas respecto al total de créditos, los bachilleres admitidos al programa tendrán que completar cierta cantidad de cursos adicionales. Estos estudiantes deben tomar créditos adicionales en cursos de investigación y deben aprobar 5 cursos en las áreas de *Análisis de Algoritmos*, *Sistemas Operativos Avanzados*, *Bases de Datos*, *Lenguajes de Programación* y *Redes de Computadores*. Estos cursos, aunque obligatorios, no forman parte del conjunto de cursos ofrecidos por el Programa de Doctorado, sino que deben ser cubiertos por el estudiante tomando y aprobando cursos de estos temas en los Programas de Maestría en Computación de las universidades públicas asociadas al Doctorado. Nótese que no es indispensable que dichos cursos sean enseñados por profesores con grado mínimo de doctorado. La selección de cursos a ser tomados para cubrir este requisito deberá ser aprobada, caso por caso, por el *C.E.D.*. De la misma forma, es también potestad del *C.E.D.* decidir si los cursos tomados son o no equivalentes a los 20 créditos requeridos en este rubro.

6 Perfil del Graduado

Un "doctor" es, esencialmente, un científico. Debe tener un dominio amplio, profundo, riguroso y actualizado de su disciplina. Al mismo tiempo, debe estar entrenado como investigador para que cumpla su misión fundamental: crear conocimiento nuevo. Un doctor está capacitado para hacer crecer, depurar y refinar el conjunto de conocimientos que llamamos *Ciencia*. Es clave para el Programa propuesto el desarrollo del pensamiento científico, complejo y disciplinado de alto nivel que le permita a los graduados manejar con rigurosidad y objetividad los procesos científicos inherentes a la investigación, así como los procesos sociales y empresariales atinentes a su desarrollo. Los graduados estarán entrenados para el desarrollo de artículos científicos, propuestas de investigación, y publicaciones técnicas en general. Serán capaces de identificar y evaluar problemas de investigación y desarrollo, así como planear estrategias de trabajo para su solución. Estarán capacitados para organizar y dirigir grupos de trabajo que realicen investigación original y de frontera, tanto en el ámbito científico como en el de las industrias y organismos vinculados a la problemática de la Computación e Informática. El graduado del Programa de Doctorado también estará en capacidad de impartir lecciones de carácter general o específico, utilizando las herramientas y medios de información más modernos, a nivel de Bachillerato, Maestría o Doctorado.

El graduado podrá plantear y buscar soluciones de la mayor calidad en forma rápida y eficiente de los distintos problemas que se presenten, en su área de énfasis profesional. Así mismo, su capacidad para crear soluciones le permitirá dar nuevos aportes o desarrollar nuevos esquemas tecnológicos. También, se espera que el graduado adquiera valores y actitudes que lo identifiquen como un profesional con alto nivel ético con respecto al ambiente, la equidad de género, la justicia y la solidaridad.

En detalle, el graduado del Doctorado en Ciencias de la Computación, será altamente competente para desempeñarse en las siguientes funciones:

- *Diseñar, ejecutar y evaluar programas de investigación en el campo de las Ciencias de la Computación. Promover el establecimiento de estos programas, impulsando y controlando procesos de investigación bajo los más estrictos principios de sistematicidad, rigurosidad y objetividad.*
- *Dirigir Centros y equipos de investigación en el campo de la Computación e Informática.*
- *Publicar, de acuerdo a las normas del idioma y de las publicaciones en el campo científico, los resultados de sus investigaciones. También comunicar a auditorios variados los resultados de estas investigaciones.*
- *Promover en sus pares y profesionales de otras disciplinas el placer y la satisfacción de buscar la verdad en forma sistemática, rigurosa y objetiva.*
- *Ejercer la docencia a cualquier nivel (grado o postgrado) en el campo de Ciencias de la Computación y disciplinas afines.*
- *Solucionar problemas concretos, desde el campo de las ciencias de la computación, en los procesos administrativos e industriales.*
- *Plantear y promover el establecimiento de convenios con organizaciones nacionales e internacionales en temas relacionados a la disciplina de Computación e Informática.*

Se tiene muy claro el compromiso que el Programa de Doctorado debe tener con una serie de principios, no sólo por correspondencia a diversas normativas, sino y sobre todo, por la importancia de darle este carácter distintivo al

graduado que tanto necesita la sociedad. El graduado del programa poseerá las siguientes actitudes, valores y principios:

- *Respeto a la disciplina científica, sus reglas, procedimientos, y sobre todo respeto a la búsqueda de la verdad.*
- *Alto grado de criticidad y creatividad.*
- *Manejo respetuoso hacia la dignidad de las personas.*
- *Sensibilidad por los problemas sociales y ambientales.*
- *Apertura permanente hacia las nuevas ideas y situaciones sociales.*
- *Capacidad para trabajar en entornos de alta competitividad.*
- *Tolerancia hacia las diferencias culturales y personales.*
- *Objetividad y responsabilidad en el cumplimiento de sus funciones.*

7 Resumen

Las universidades estatales de Costa Rica se han propuesto el desarrollo de un programa conjunto para la formación de Doctores en Ciencias de la Computación. Se reconoce la importancia fundamental de la investigación científica en dicho programa (evaluada bajo las normas de *publish or perish*) y se crea junto con el Programa de Doctorado el Centro de Investigación en Computación e Informática Avanzada (*CIenCIA*). Al mismo tiempo, se exige que este programa tenga calidad de nivel mundial. Durante las fases de desarrollo de esta propuesta hemos recibido retroalimentación internacional de representantes de universidades de Latinoamérica, Estados Unidos y Europa que han resultado invaluables para mejorar nuestro intento.

Hay consenso en la comisión universitaria en la búsqueda de la excelencia y la prioridad indudable de la investigación científica en el Programa de Doctorado. Sin embargo, la discusión en cuanto a los medios para lograr este fin ha sido muy interesante y enriquecedora. Esta discusión tiene cómo base la pugna entre el sistema europeo y el sistema norteamericano de estudios de doctorado, y el constante cuestionamiento de la pertinencia, ideoneidad y factibilidad de ambos esquemas en el ámbito costarricense.

Aunque el programa aún está por empezar, los resultados hasta este punto nos hacen sentir muy optimistas respecto a la factibilidad del mismo. Hemos logrado una excelente comunión de ideas entre los representantes de las universidades, respetando las idiosincrasias de cada universidad (e inclusive las de la mayoría de los profesores involucrados) pero ofreciendo, al mismo tiempo, un proyecto conciso y coherente. Hay amplia voluntad política de las autoridades en las universidades para que el programa se lleve acabo. Hemos recibido múltiples manifestaciones y ofrecimientos de apoyo financiero y logístico de agencias internacionales y de representantes importantes del sector industrial costarricense. También, ya sabemos de muchos estudiantes potenciales (costarricenses y de la región centroamericana) con gran interés por ingresar a este Doctorado.

Referencias

1 **Becker, Howard**, "*Tricks of the Trade - How to think about your Research While Doing It*", The University of Chicagp Press, 1998.
2 **Bloom, D., Karp, J.D., Cohen, N.**, "*The Ph.D. Process: A Student's Guide to Graduate School in the Sciences*", 2002.
3 **Bolker, Joan**, "*Writing your Dissertation in 15 minutes a day: A Guide to Starting, Revising and Finishing your Doctoral Thesis*", 1998.
4 **Booth, Wayne et al.**, "*The Craft of Research*", The University of Chicago Press, 1995.
5 **Davis, G., Parker, C.**, "*Writing the Doctoral Dissertation: A Systematic Approach*", 1997.
6 **Feibelman, Peter J.**, "*A Ph.D. is not Enough: A Guide to Survival in Science*", 2001.
7 **Feynman, Richard**, "*The Meaning of It All*", Addison Wesley, 1998.
8 **Feynman, Richard**, "*The Pleasure of Finding Things Out*", Perseus Books, 1999.
9 **Marín, G., Torres-Rojas, F.**, "*Doctorado Nacional en Ciencias de la Computación en Costa Rica: cooperación interuniversitaria en pos de la investigación*", VII Junta Consultiva sobre el Posgrado en Iberoamérica, Convención Universidad 2004, La Habana , Cuba, Febrero, 2004.
10 **Peters, Robert,** "*Getting What You Came For - The Smart Student's Guide to Earning a Master's or Ph.D.*", The Noonday Press, Revised Edition, 1997.
11 **Pugh, D.S., Phillips, E.,** "*How to Get a Ph.D.: A Handbook for Students and Their Supervisors*", 2002.
12 **Rudestam, Kjell Erik and Newton, Rae R.,** "*Surviving your Dissertation: A Comprehensive Guide to Contens and Process*", 2002.
13 **Smith, Robert V.**, "*Graduate Research: A Guide for Students in the Sciences*", 1999.

14 **Torres-Rojas, F., Marín, G.**, "*Propuesta para un Programa Nacional de Doctorado en Ciencias de la Computación en Costa Rica*", V Congreso Chileno de Educación Superior en Computación, CCESC-03, XI Jornadas Chilenas de Computación, Chillán, Chile, Noviembre, 2003.

15 **Turabian, Kate L.**, "*A Manual for Writers of Term Papers, Theses and Dissertations*", The University of Chicago Press. Sixth Edition, 1996

Aprendizaje Orientado por Proyectos: Una Aplicación en los Cursos de Ingeniería de Software.

Abraham E. Dávila Ramón
Pontificia Universidad Católica del Perú, Facultad de Ciencias e Ingeniería, Grupo de Investigación y Desarrollo en Ingeniería de Software.
Lima, Perú, Lima 32
edavila@pucp.edu.pe

Abstract

The Project Oriented Learning (POL) is a didactical technique that it is used in our software engineering courses at Informatic Engineering School. We have use POL for many years ago and its benefits are richer than only acquire knowledge. In this paper, show the educational innovation have been doing in our courses since these courses taught first time until nowadays, in particular, describe the educational model used in our software engineering courses today.
Keywords: Project Oriented Learning, Education, Software Engineering, Teaching Methods.

Resumen

El Aprendizaje Orientado por Proyectos es una técnica didáctica que se viene empleando en los cursos del área de Ingeniería de Software de la Especialidad de Ingeniería Informática desde hace varios años y cuyos beneficios van más allá de la adquisición de conocimiento. En el presente artículo, se presenta la innovación pedagógica realizada de manera continua desde su introducción hasta la fecha, y en particular, se describe el actual modelo didáctico empleado en los cursos de ingeniería de software.
Palabras claves: Aprendizaje Orientado por Proyectos, Educación, Ingeniería de Software, Modelo Didáctico.

1. INTRODUCCIÓN

En el año 1990 se crea la especialidad de Ingeniería Informática en la Facultad de Ciencias e Ingeniería (FCI) de la Pontificia Universidad Católica del Perú (PUCP) [8]. La primera vez que se dictó el curso de Ingeniería de Software y los cursos-taller complementarios (desarrollo de programas 1 y desarrollo de programas 2) se empleó un modelo didáctico diferente del resto de cursos. El modelo didáctico por un lado se basó en un equipo de estudiantes que desarrollan un proyecto de software desde la comprensión del problema hasta su total implementación; cubriendo todas las etapas usuales de un proyecto real de desarrollo de software, y que por otro lado, se complementa con las sesiones teóricas correspondientes.

La forma en que se dictó por primera vez y la forma en que actualmente se dictan todos los cursos obligatorios en el plan de estudios de la línea de ingeniería de software, ha variado de manera significativa principalmente por las experiencias docentes que se ha obtenido en todos estos años en las sucesivas innovaciones introducidas. Los proyectos que desarrollan los estudiantes, durante el semestre académico, tienen diferentes características entre curso y curso, y buscan en cada caso, emular situaciones reales de los proyectos de desarrollo de software considerando las restricciones propias del ámbito académico y buscando enfrentar a los estudiantes a diversas situaciones tal como ocurren en la vida profesional.

En el año 2000, se presentó a todos los miembros de la comunidad universitaria de la PUCP, el Plan Estratégico Institucional (PEI) 2000-2010 [9]; en él se estableció, como una meta, la introducción de metodologías que mejoren la actividad educativa en general; hasta ese momento la mayoría de los cursos se basaban en un modelo didáctico tradicional (sesiones magistrales). Tomando como marco de referencia el PEI y luego de una evaluación, por parte de la Comisión de Modernización Pedagógica (CMP) del Vicerrectorado Académico, se optó por introducir métodos activos y colaborativos [10], siendo uno interesante la técnica didáctica ABP de Aprendizaje Basado en Problemas (también conocido como BPL de sus siglas en inglés de Based Problem Learning). La corriente que se inició entre los docentes por introducir ese modelo didáctico, nos ha llevado a ahondar y descubrir que el modelo que hemos seguido desde nuestros inicios se denomina Aprendizaje Orientado por Proyectos (AOP) y que es más conocido como POL (por sus siglas en inglés de Project Oriented Learning). AOP es una técnica didáctica muy cercana al ABP y la frontera entre uno y otro es una tenue línea que a veces no se distingue [7]. Nosotros lo optamos pues sentimos que era una manera natural de aprender a administrar los proyectos de desarrollo de software.

Nuestro actual modelo didáctico se encuentra adaptado a cada uno de los cursos que comprenden la línea del área de ingeniería de software y considera las diversas variables como: número de estudiantes por equipo, tipo de proyecto de software, cantidad de entregables del proyecto y conocimientos obtenidos en los cursos.

En este documento, revisaremos brevemente los modelos didácticos de ABP y POL, luego se revisará el dictado de los cursos de la línea de ingeniería de software desde sus inicios y los cambios sucesivos hasta la actualidad. Finalmente se presentará una evaluación cualitativa y las conclusiones y cambios futuros a introducir en esta línea de trabajo.

2. APRENDIZAJE BASADO EN PROBLEMAS.

La técnica ABP tiene sus primeras aplicaciones en la década de los 60's en la escuela de medicina de la Universidad de Case Western Reserve en EEUU y en la Universidad de McMaster en Canada [4] planteándose situaciones de la vida real como un mecanismo integrador de los conocimientos necesarios –de diversas áreas del conocimiento- para resolver el problema.

ABP se define, según [3], [4], como un método didáctico de aprendizaje en el que grupos pequeños resuelven un problema propuesto por el docente y en cuya resolución se logra aprendizajes significativos. Los beneficios del uso del ABP se pueden resumir según [3] en los siguientes: (i) pensar críticamente y ser capaz de analizar y resolver problemas complejos de la vida real; (ii) encontrar, evaluar y utilizar las fuentes de información adecuadas; (iii) trabajar cooperando en equipos y grupos pequeños; (iv) mostrar habilidades versátiles y eficaces de comunicación, tanto verbalmente como por escrito; y (v) usar el conocimiento de los contenidos y las habilidades intelectuales adquiridas en la universidad para convertirse en permanentes estudiantes (que pueden aprender permanentemente).

3. APRENDIZAJE ORIENTADO POR PROYECTOS.

Uno de los primeros trabajos de AOP -no como una técnica didáctica propiamente- se presenta en el trabajo de Kilpatrick sobre "Desarrollo de Proyectos" en 1918 según [5] y cuya visión global abarca el proceso completo del pensamiento desde la idea inicial hasta la solución completa del problema.

AOP se define, según [5],[12],[6],[2], como un método didáctico de aprendizaje en el que los estudiantes toman una mayor responsabilidad de su propio proceso de aprendizaje aplicando en un proyecto sus habilidades y conocimientos adquiridos previamente. Sin embargo, según [11], para determinar si un proyecto es o no de AOP, se debe de cumplir: (i) los proyectos son componentes centrales y no periféricos al currículum; (ii) los proyectos se enfocan en problemas que inducen a los estudiantes a enfrentarse a los conceptos y principios básicos de una o varias disciplinas; (iii) los proyectos implican a los estudiantes en un proceso de investigación creadora; y (iv) los proyectos son dirigidos, en gran medida, por los mismos estudiantes.

Uno de los trabajos más referenciado sobre AOP es el de Blumenfeld y otros [1] en donde señalan, entre otras cosas, que los estudiantes resuelven problemas diversos y complejos para completar su proyecto al: (i) hacer y depurar preguntas; (ii) diseñar planes y/o experimentos; (iii) hacer predicciones; (iv) recolectar y analizar datos; (v) debatir ideas; (vi) establecer conclusiones; (vii) comunicar ideas; (viii) hacer nuevas preguntas; y (ix) crear artefactos. También resaltan que el proyecto es de una duración relativamente larga, centrada en un problema y que involucra conceptos de otras disciplinas y campos de estudio.

Usar AOP en un curso definitivamente introduce cambios en todos los participantes. Algunos de ellos, como lo señalan en [5],[13],[1],[11], están referidas: a la relación entre los profesores y los estudiantes, al cambio de actitud de competencia entre ellos hacia colaboración entre sus pares y de la memorización de contenidos al aprendizaje real pues lo necesitarán para resolver su problema.

4. DESCRIPCIÓN DE LOS CURSOS DEL ÁREA DE INGENIERÍA DE SOFTWARE.

Los cursos del área de Ingeniería de Software que son materia de este artículo son: Ingeniería de Software, Desarrollo de Programas 1 y Desarrollo de Programas 2. Ellos son parte de la columna vertebral en el último año y medio de estudios en nuestra especialidad y cada uno es un requisito para el siguiente.

Ingeniería de Software (IS) es el primer curso en donde se presentan metodologías de desarrollo de software y manejo de proyectos. Los estudiantes trabajan de manera paralela un proyecto de construcción de software acorde al nivel de los estudiantes durante todo el periodo lectivo. Las sesiones de teoría se llevan a cabo en aulas de clases y la supervisión del proyecto (sesiones de práctica del curso) en los laboratorios. Existe correspondencia entre lo que se dicta en clase y lo que se desarrolla en el proyecto.

Desarrollo de Programas 1 (DP1) es el primer curso de integración de conocimientos de todos los cursos previos. Se da un mayor énfasis a los temas de verificación y validación, gestión de la configuración, métricas e indicadores, entre otros. Los estudiantes trabajan un proyecto durante todo el periodo lectivo. Las sesiones teóricas se desarrollan principalmente en la primera parte del curso y durante todo el curso se supervisa y apoya el desarrollo del proyecto.

Desarrollo de Programas 2 (DP2) es el segundo curso de integración de conocimientos de la línea de Ingeniería de Software, que permite la integración de conocimientos y experiencias para el desarrollo de una solución de tecnologías de información en el dominio de los sistemas de información. Los estudiantes trabajan el proyecto durante todo el periodo lectivo. No hay sesiones teóricas, los estudiantes organizan las sesiones técnicas de acuerdo a las necesidades propias del proyecto y del equipo de desarrollo.

5. DICTADO INICIAL DE CURSOS DEL ÁREA DE INGENIERÍA DE SOFTWARE.

La primera vez que se dictó IS (1994-1), se intentó dirigir un proyecto completo entre todos los estudiantes, sin embargo debido a la falta de experiencia en este tipo de situaciones se tuvo que recortar algunas funcionalidades del sistema de información solicitado. Los estudiantes se entusiasmaron y construyeron el producto; fue una buena experiencia para ellos por que era su primer proyecto de principio a fin, es decir, desde la identificación de necesidad

del problema hasta su implantación. El contenido teórico fue revisado rápidamente y pasado a un segundo nivel de importancia por parte de los estudiantes.

La primera vez que se dictó DP1 (1994-2) se conformaron equipos pequeños para desarrollar software de acuerdo a los intereses de los propios estudiantes. La experiencia fue interesante pero la mayor dificultad era determinar una balance adecuado de la carga de trabajo entre los estudiantes.

La primera vez que se dictó DP2 (1995-1) se planteó la continuación de los trabajos inicialmente desarrollados en el curso previo; que permitió a los estudiantes completar ideas de productos interesantes y mejorar la versión anterior.

6. CAMBIOS EN LA PRÁCTICA DOCENTE EN LOS CURSOS DE INGENIERÍA DE SOFTWARE.

Los cambios en la práctica docente se fueron dando de acuerdo a las modificaciones del plan de estudios. Los cambios más significativos y en que cursos se efectuaron se presenta en la tabla 1 y tabla 2. Se han separado en dos tablas para dar énfasis a un cambio de contenidos radical que se produce en el periodo lectivo 2002-2 en los cursos IS y DP2.

Periodo	Curso	Cambios introducidos
1994-1	IS	Se dicta por primera vez el curso. Los estudiantes desarrollan un único proyecto trabajando en equipos.
1994-2	DP1	Se dicta por primera vez el curso. Los estudiantes en equipos pequeños desarrollan un proyecto.
1995-1	DP2	Se dicta por primera vez el curso. Los estudiantes en equipos pequeños continúan el proyecto que desarrollaron en el curso previo.
1996-1	DP1	Se introduce la lista de exigencias como herramientas para controlar el entregable final. La lista de exigencias es un catálogo de requisitos de donde los estudiantes identificaban aquellos requisitos de producto que al final tendrían que ser implementados por ellos. La lista de exigencias resultó ser practico pues los estudiantes y profesores podían negociar convenientemente el alcance del proyecto a desarrollar.
1996-2	DP1	Se introduce el uso de la evaluación de desempeño. Una herramienta para reportar el desempeño de los demás miembros del equipo; se empleó la valoración vigesimal de manera análoga a como se evalúa en el curso. La evaluación de desempeño se utilizó para identificar potenciales problemas entre los equipos y poder supervisar desde adentro el desarrollo del proyecto. Las reuniones semanales con los asistentes de docencia y los docentes complementaba la evaluación de desempeño.
1996-2	DP2	Se introduce el uso de la lista de exigencias. Los estudiantes desarrollan un proyecto en equipos pequeños, pero diferente al curso anterior y diferentes para cada equipo. Se definen una lista de diversos proyectos de donde los estudiantes elegían cual llevarían.
1997-1	DP1	Se definen trabajos con énfasis en la parte de computación (algorítmica). Además se define que al final del curso habrá una evaluación comparativa entre todas las soluciones (un ranking) que permita estimular las mejores soluciones. Se introduce una actividad de "auditoria" en el diseño. Todos los grupos rotan sus trabajos y tienen que informar si la documentación (del diseño) hasta ahora elaborada es posible de ser implementada. Los estudiantes se dan cuenta que la documentación es muy importante y exponen las deficiencias en clase.
1997-1	DP2	Se introduce la evaluación de desempeño cerrada (los estudiantes reportan evaluación de sus pares).
1998-1	DP1	Se cambia el criterio de uso en la evaluación de desempeño (solo en un horario de clases).
1999-1	DP1	Se introduce la co-evaluación y la evaluación de desempeño abierta al final de periodo. Se les pide a los estudiantes que hagan una evaluación sobre si mismo primero para que puedan fijar su nivel de referencia y luego evalúen al resto de su equipo de trabajo. Se definen cinco criterios de evaluación: conocimiento, puntualidad, motivación, responsabilidad, proactividad. La evaluación de desempeño se reportará por correo electrónico a los asistentes de docencia semanalmente.
1999-1	DP2	Se introduce la co-evaluación y la evaluación de desempeño abierta al final de periodo.

Tabla 1 Cambios en el periodo 1996-1 al 2000-1

Periodo	Curso	Cambios introducidos
2000– 2	IS	Se cambia el esquema de trabajo: se introduce el concepto de empresas de desarrollo y 3 frentes de desarrollo. Empresas con alrededor de 10 estudiantes compuesto de 3 frentes de 3 estudiantes en promedio.
2000– 2	DP1	Se formaliza el tamaño de equipo a 3 ó 4 estudiantes.
2000– 2	DP2	Se cambia el esquema de trabajo: se trabaja con una sola empresa por curso-horario que tiene la responsabilidad de desarrollar un sistemas de información tipo ERP con 6 grandes módulos. El número de estudiantes por horario es alrededor de 30 estudiantes.
2002– 2	IS	Se establece que se empleará Delphi y MSSql como herramientas de desarrollo de software.
	DP1	Se establece que el trabajo será desarrollado con Java en un esquema Cliente-servidor. Se usará XML para repositorio de información en los casos que se requieran guardar archivos.
	DP2	Se establece que se trabajará con Java en Web y usando un RDBMS de software libre (Postgresql).
2003– 1	todos	Se introduce un esquema de definición de equipos de trabajo basado en grupos parciales formados por el profesor a partir de los rendimientos académicos de los estudiantes; de modo que los equipos queden relativamente balanceados en capacidad de los alumnos.
2004– 1	IS	Se cambia la frecuencia de la evaluación de desempeño a quincenal y se cambia la valoración a percepciones de cumplimiento usando una escala de 0 a 10.

Tabla 2: Cambios en el periodo 2000-2 al 2004-1

7. LOS CURSOS DE INGENIERÍA DE SOFTWARE.

En la especialidad de informática se da énfasis a la parte algorítmica y de técnicas de programación en los primeros semestres. Luego los estudiantes toman cursos de modelamiento de sistemas de información estructurados y orientados a objetos y llevan el curso de bases de datos. Adicionalmente, en forma paralela, los estudiantes llevan el curso de sistemas operativos y otros de áreas complementarias. Los estudiantes ya tienen todas los conocimientos básicos necesarios para poder ejecutar un proyecto informático. El curso de IS brinda conocimientos teóricos sobre ingeniería de requisitos, manejo de proyectos (usando el PMBOK de PMI www.pmi.org), ciclos de vida de software, gestión de riesgos, calidad de software, estimación de software, entre otros y es el primer curso donde hacen un proyecto completo de software en equipos de trabajo desde el planteamiento hasta su implementación.

El curso de IS se dicta en el 8 nivel de estudios, el curso de DP1 se ubica en el nivel 9 y DP2 se ubica en el nivel 10. Cada uno es requisito del otro, pues consideramos que es un proceso natural de evolución y maduración del estudiante en el tema de manejo de proyectos. Las características de los cursos que actualmente se dictan en el área de ingeniería de software se presentan en las siguientes tablas resúmenes. La tabla 3, se presenta la características con un fuerte énfasis en el eje tecnológico.

Aspecto del Proyecto	Ingeniería de Software	Desarrollo de Programas 1	Desarrollo de Programas 2
Área Informática	Sistemas de Información	Ing. de computación / Ciencias de la computación.	Sistemas de información
Tipo de software	Cliente / servidor	Variado	Aplicación web
Lenguajes de Programación	Delphi, Object Pascal	Java	Java
Sistema Administrador de Bases de datos	MS-Sql Server	No usa RDBMS, sino XML, en caso requiera almacenar información	Postgresql o MySql
Arquitectura de Software	Cliente / Servidor	Elegido por los estudiantes	Tres capas
Proyecto del periodo 2002-2.	Sistema de compra, armado de paquetes y asistente de paquetes turísticos.	Software para la generación de datos de pruebas usando reglas de construcción.	Sistema para la Planificación de Recursos Empresariales (ERP).
Proyecto del periodo 2003-1.	Sistema de compra, venta y almacenamiento de una tienda de videos.	Software para la prueba de esfuerzo de aplicaciones desarrolladas en Java.	Sistema Integrado para Gobiernos Locales y Regionales.

Tabla 3: Características de los cursos de Ingeniería de Software en el eje técnico.

Las decisiones de porque cada curso cubre un tema diferente fueron acordados en reuniones de coordinación de los profesores del área. En el caso concreto de las herramientas de desarrollo empleadas, se han definido las que aparecen en el cuadro, pues en los cursos previos se ven otras herramientas y la idea es que el estudiante tenga conocimiento práctico en las que más se usan en la industria nacional. Por ejemplo el Sistema Administrador de Bases de Datos que se utiliza en el curso de Bases de Datos es Oracle, por lo que ya no se le considera en este esquema.

Aspecto del Proyecto	Ingeniería de Software	Desarrollo de Programas 1	Desarrollo de Programas 2
Total estudiantes	30	30	30
Estudiantes por empresa	10	3 a 4	30
Estudiantes por frente de trabajo	3 a 4	No aplica	6
Desarrollo de cada frente	Componente funcional.	No aplica	Componente funcional.
Coordinación interna	Sí	No	Sí
Comité de estándares	Sí	Sí	Sí

Tabla 4: Características de los cursos de Ingeniería de Software en el eje organización de equipos.

La tabla 4, presenta la información sobre los equipos de desarrollo que se conforman durante los cursos. Se emplea el concepto de empresa como principal agrupador (agrupador de mayor nivel) y el concepto de equipos de trabajo (frentes) para un siguiente nivel de agrupamiento. Por la naturaleza de cada proyecto, en algunos casos, es necesario establecer comités o coordinaciones que faciliten el tema de integración del software en desarrollo en sus distintas etapas; estas asociaciones son eventuales y en la medida de lo posible es conformado por un miembro de cada frente. El número de estudiantes por empresa y por frente es un valor referencial, y se definen en cada curso horario de acuerdo al total de inscritos.

La tabla 5, presenta la forma de participación del equipo docente en este tipo de cursos. Desde hace dos años establecimos que los asistentes de docencia tuvieran una experiencia profesional en manejo de proyectos mayor a 2 años como egresados de la universidad. Este requisito se ha ido cumpliendo paulatinamente y actualmente se cumple para todos los casos. La experiencia cuenta mucho al momento de trabajar con los estudiantes.

Aspecto del Proyecto	Ingeniería de Software	Desarrollo de Programas 1	Desarrollo de Programas 2
Nivel de participación de los asistentes de docencia en el proyecto.	Alta, guían –sin hacerles la solución- a los equipos de trabajo en sus decisiones. Influyen en las decisiones. Actúan como asesores del equipo.	Media, cuestionan las decisiones y ofrecen varias alternativas, pero se insiste que es el equipo quien asume la responsabilidad de la decisión. Actúan como asesores.	Baja, cuestionan todas las decisiones funcionales y técnicas. Actúan como auditores.
Nivel de participación de los docentes en el proyecto.	Supervisa a los equipos, a los asistentes de docencia y negocia los alcances del proyecto. Revisa y acuerda el plan de trabajo durante 3 diferentes momentos del proyecto.	Supervisa y negocia el alcance del proyecto al inicio y a la mitad del periodo. Introduce cambios en el escenario de desarrollo para que los equipos reaccionen ante las eventualidades.	Supervisa continuamente el desarrollo del proyecto y el alcance del mismo. La fecha de entrega es definida por la empresa.

Tabla 5. Características de los cursos de Ingeniería de Software en el eje de los docentes.

Los requisitos establecidos para poder llevar el curso de Ingeniería de Software, que es el primero de la línea, son los siguientes: (i) Sistemas de información 2 (modelado y diseño orientado a objetos con el Lenguaje Unificado de Modelado UML), (ii) Lenguaje de Programación 2 (Programación orientada a objetos), (iii) Bases de datos (diseño de bases de datos) y (iv) cursos de administración. Todo estos conocimientos previos permite que el curso se oriente más a la parte de gestión del proyecto de software y a los temas en procesos, metodologías, requisitos y calidad.

Los cursos definidos y manejados de la forma ya descrita, cumplen con los criterios establecidos por Thomas[11] para ser considerados del tipo AOP y el resumen se presenta en la tabla 6 y se aprecia las características señaladas por Blumenfeld [1] en la tabla 7. Los proyectos tienen una duración promedio de 15 semanas y demandan el conocimiento del dominio del problema.

Criterios para determinar si es AOP según Thomas	Ing.Sw	DP1	DP2
El proyecto es un componente central y no periférico	X	XX	XX
Los problemas inducen a enfrentamiento a los conceptos y principios ...	X	XX	X
El proyecto implica un proceso de investigación creadora.	X	X	X
El proyecto es dirigido principalmente por los alumnos.	X	XX	XXX

Tabla 6. Cumplimiento de criterios para ser considerados cursos AOP

La tabla 6, muestra los niveles de cumplimiento de cada criterio. La diferencia en el grado de cumplimiento se debe a que en IngSw existe más influencia del equipo docente sobre varios aspectos como la planificación maestra, la definición del proceso seguido y la supervisión más cercana de las decisiones. En cambio en el otro extremo está DP2, donde los alumnos toman el control total de todo el proyecto y el único dato de referencia es la fecha final del proyecto, todo lo demás lo hacen ellos. Con respecto al criterio de los problemas inducen al enfrentamiento con los conceptos, se nota más en DP1 porque el problema usualmente así lo amerita, al ser un problema más computacional las herramientas de modelado de sistemas de información no les resulta del todo adecuadas y ellos deben resolver dicha situación.

Conducta de alumnos observados en el caso de AOP según Blumenfeld.	Ing.Sw	DP1	DP2
Hacer y depurar preguntas	X	X	X
Diseñar planes y/o experimentos	X	XX	XXX
Hacer predicciones /estimaciones	X	X	XX
Recolectar y analizar datos	X	X	X
Debatir ideas	XX	X	XXX
Establecer conclusiones	X	XX	XX
Comunicar ideas	X	XX	XX
Hacer nuevas preguntas	X	X	X
Crear artefactos.	X	XX	XXX

Tabla 7. Conductas observadas en los cursos de AOP.

Las conductas observadas y resumidas en la tabla 7 están en correspondencia con las características propias de los cursos. En DP2 se aprecia con mayor énfasis las conductas observadas en la aplicación de AOP, pues los estudiantes tienen un control mayor del proyecto en si.

8. EL PROCESO DE SOFTWARE.

Luego de trabajar con diversos alternativas de ciclo de vida en los cursos, se optó por que en el curso de Ingeniería de Software se use un ciclo de vida de proyecto basada en dos iteraciones en la parte de construcción. En la figura 1 se muestra el ciclo de vida establecido por los profesores para dicho curso. En los cursos de DP1 y DP2 los estudiantes tienen la responsabilidad de definir el ciclo de vida de sus proyectos.

9. EVALUACIÓN EN LOS CURSOS DE INGENIERÍA DE SOFTWARE.

Antes de hablar sobre la evaluación a los estudiantes, es importante señalar, que los proyectos se desarrollan en aproximadamente 15 semanas de trabajo y tomando en cuenta la carga académica de los estudiantes, es correcto pensar que el producto final dista funcionalmente de un producto real. Cada curso, impone ciertas restricciones en la parte funcional al momento de la implementación final o establece un alcance de acuerdo a las negociaciones que se desarrollan entre profesores y equipos de trabajo.

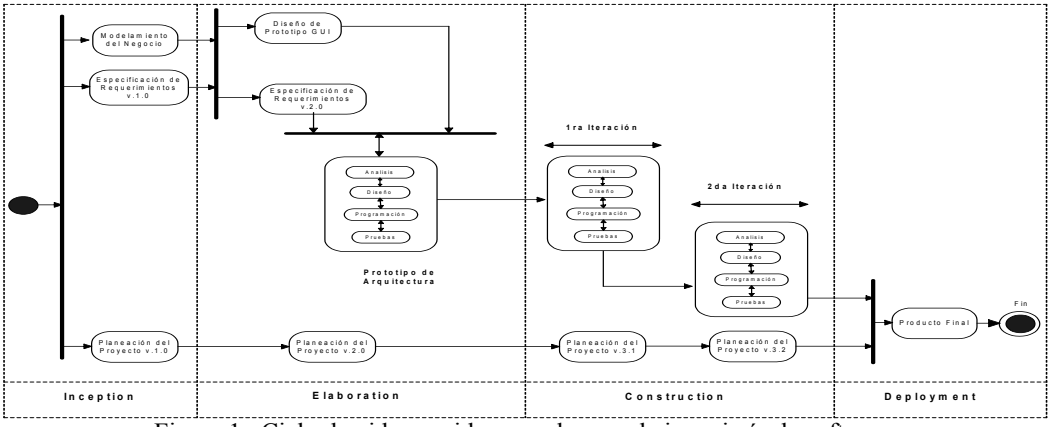


Figura 1. Ciclo de vida seguido para el curso de ingeniería de software.

La evaluación en cada uno de los cursos dependen de diversos componentes. Para cursos bajo AOP, la evaluación final depende fuertemente del producto desarrollado. Una práctica empleada es que el producto debe funcionar de manera correcta al final del periodo lectivo, si ello no ocurre entonces el estudiante no es aprobado. Si el producto funciona correctamente, entonces el estudiante, el trabajo y la gestión del proyecto son evaluados por los asistentes de docencia, el profesor del cursos y por los propios estudiantes. La evaluación se realiza durante todo el proyecto conforme se van entregando los avances, pero la calificación final queda supeditada al correcto funcionamiento del software de acuerdo a las funcionalidades establecidas.

10. CONCLUSIONES Y TRABAJO FUTURO.

Los cursos de Ingeniería de Software, Desarrollo de Programas 1 y Desarrollo de Programas 2 son ejemplos concretos de la aplicación del modelo de didáctico de Aprendizaje Orientado por Proyectos y que cumple con lo afirmado por Thomas[11] y Blumenfeld [1]. Aunque inicialmente no se partió del conocimiento de los modelos didácticos, la experiencia docente permitió derivarlos a la situación actual y que ofrece mayores beneficios según diversos autores [2],[3] y [4]. La experiencia obtenida nos está permitiendo introducir mejoras a los cursos previos.

Algunas mediciones que se están haciendo en el uso de ABP están arrojando resultados alentadores sobre los beneficios en su aplicación. De manera análoga, apoyados en una encuesta abierta –todavía- realizada sobre un grupo de egresados, comentan que el manejo de las diversas situaciones que se enfrentan en los cursos suelen ser similares a los que viven en las empresas actualmente y que les ha servido el haberlo vivido previamente para tomar decisiones.

La manera de evaluar a los alumnos ha tenido necesariamente que evolucionar desde los modelos tradicionales hacia modelos más convenientes. Un caso interesante se dan por el lado de la AutoEvaluación y la Co-Evaluación de los equipos de estudiantes sobre el producto desarrollado y sobre la gestión del desarrollo, en ella los estudiantes asumen una posición crítica sobre su trabajo y la de sus compañeros. Sobre la parte de evaluación de desempeño de los equipos de trabajo se está desarrollando una herramienta informática que apoye esta actividad.

Aplicar AOP a estos cursos resulta muy conveniente pues se trata de cursos integradores de conocimientos ubicados en los últimos semestres de la carrera. Sin embargo demanda un tiempo mayor por parte del equipo docente quienes se apoyan –como resulta obvio- en todas las bondades que ofrecen hoy las tecnologías de la información. El equipo docente (profesores y asistentes de docentes) trabajan relativamente más que en los cursos tradicionales (según lo que se observa), sin embargo el trabajo es más estimulante porque todos aprenden durante la ejecución del proyecto. Con apoyo del Centro del Magisterio Universitario (MAGIS-PUCP) se realizarán algunas mediciones sobre el uso de está técnica didáctica.

Agradecimiento

Deseo expresar mi agradecimiento a Silvana Díaz que preparó el diagrama de actividades para un trabajo interno de la especialidad, a Carla Basurto por las sugerencias y a Luis Bretel por animarme a escribir este artículo y por revisarlo.

Referencias

[1] Blumemfeld, P., Soloway, E., Marx, R., Krajcik, J., Guzdial, M., Palincsar A., Motivating Project-Based Learning: Sustaining the Doing, Supporting the Learning. Educational Psychologist, 26(3&4), 369-398. Lawrence Erlbaum Associates, Inc, 1991.

[2] Bucciarelli, L. Project Oriented Learning as Part of Currículo Development. 3rd NTVA Industrial Ecology Seminar and Workshop. 1998. http://www.indecol.ntnu.no/indecolwebnew/events/conferences/ntva/3rd_ntva/3rd_ntva.htm

[3] Duch , B., Groh, S., Allen, D., ¿Por qué el Aprendizaje Basado en Problemas? Un Estudio de Casos del Cambio Institucional en la Educación de Pregrado. El poder del Aprendizaje basado en Problemas. Editado por Duch , B., Groh, S., Allen, D. (en proceso de publicación 2004).

[4] ITESM, El Aprendizaje Basado en Problemas como Técnica Didáctica. Dirección de Investigación y Desarrollo Educativo. Material de Curso. Vicerrectoria Académica, Instituto Tecnológico y de Estudios Superiores de Monterrey. 2003.

[5] ITESM, El método de proyectos como técnica didáctica. Dirección de Investigación y Desarrollo Educativo. Material de Curso. Vicerrectoria Académica, Instituto Tecnológico y de Estudios Superiores de Monterrey. 2003.

[6] Meyer, V, Project Oriented Learning (POL) as a Comunication Tool of Enviromental Sciences in the Community of Sohanguve – A Case Study. 7th International Conference on Public Communication of Science and Technology. 2002

[7] Oakey, J. Project-Based and Problem-Based: The same or different?. http://pblmm.k12.ca.us/PBLGuide/PBL& PBL .htm [13/02/2004 08:30:51 p.m.]

[8] Peralta, O. La Sección Ingeniería Informática. *Ingeniería en la PUCP*. Año 1. No 1, pp 37-39. 1999

[9] PUCP. Plan Estratégico Institucional. http://www.pucp.edu.pe/dape/ . Dirección Académica de Planeamiento y Evaluación. [16/04/2004 06:13:45 p.m.]

[10] PUCP. Sobre la Comisión de Modernización Pedagógica. http://www.pucp.edu.pe/cmp/ . Comisión de Modernización Pedagógica. [18/04/2004 02:32:29 p.m.]

[11] Thomas, J., A Review of Research on Project-Based Learning, http://www.autodesk.com/foundation. March 2000.

[12] Vélez de C, A. Aprendizaje Basado en Proyectos Colaborativos en la Educación Superior. Brasilia: IV Congreso RIBIE.1998. http://phoenix.sce.fct.unl.pt/ribie/cong_1998/trabalhos/190m.pdf

[13] Zahava, S., Polak S., An Organizer for Project-Based Learning and Instruction in Computer Science. Proceeding of the 4th annual SIGCSE/SIGCUE ITICSE Conference on Innovationa and Technology in Computer Science Education. 1999. pp. 88-90.

Melhorando o Entendimento de Programação usando Esquemas Conceituais em Cursos Introdutórios

Thais Helena Chaves de Castro[1], Crediné Silva de Menezes[2], Alberto Nogueira de Castro Junior[1], Rosane Santos Caruso de Oliveira[2], Maria Cláudia Silva Boeres[2]

[1]Departamento de Ciência da Computação – Universidade Federal do Amazonas (UFAM)
Av. Gal. Rodrigo O. J. Ramos 3000 – 69.077-900 – Manaus – AM – Brasil

[2]Departamento de Informática – Universidade Federal do Espírito Santo (UFES)
Av. Fernando Ferrari, s/n – 29060-970 – Vitória – ES – Brasil

{credine,rcaruso,boeres}@inf.ufes.br

{thais, albertoc}@dcc.fua.br

Abstract. *This paper discusses an experience with programming courses using identification and formal representation of programming schemes and their potential for automatic analysis. A relationship with Bloom's taxonomy has been used to support the classification of these schemes. We intend to use these tools for classification as well as feedback routing with respect to source code produced by programming students.*

Resumo. *Este artigo trata da identificação e representação formais de esquemas de programas e seu potencial para a análise e classificação automáticas. É realizada uma correlação com a taxonomia de Bloom para apoiar a classificação desses esquemas. Planeja-se usar estas ferramentas na construção de ambientes que auxiliem o professor na elaboração de orientações com respeito ao código-fonte produzido por estudantes de cursos introdutórios de programação.*

Palavras-chave*: Ensino de programação; padrões de programa; representação do conhecimento.*

1. Introdução

O aprendizado de programação de computadores constitui-se em uma das mais elaboradas tarefas cognitivas. Ao estudante do assunto é requerida uma visão precisa do problema endereçado, conhecimento de um formalismo apropriado e a habilidade de combinar os elementos adequados para construir uma solução formal. Entre os vários elementos envolvidos neste processo, destacam-se a construção de abstrações e o uso de linguagens artificiais. Estes aspectos vêm, ao longo do tempo, estabelecendo-se como um grande desafio aos professores de disciplinas de ensino introdutório de programação [Mckeown et al, 1999; Joosten et al, 1993; Giegerich et al,1999].

Já se tem observado que no aprendizado inicial de atividades produtoras de artefatos, um fator de grande valia é a possibilidade de o estudante receber críticas, comentários e recomendações sobre as soluções por ele apresentadas (*feedback*). Em

programação, essa meta é atingida parcialmente através do uso de linguagens de programação para as quais exista um ambiente de compilação/interpretação. O analisador sintático produz as críticas com respeito aos erros sintáticos e pode até sinalizar algumas inadequações, como por exemplo, a falta de inicialização de variáveis. Por outro lado, ao submeter um programa à execução o estudante pode comparar os resultados obtidos com os esperados e assim avaliar a correção do seu programa. Alguns ambientes podem ainda fazer essas comparações a partir de um plano de teste. Enfim, é possível e já se tem usado o apoio das máquinas na produção de *feedback*.

Contudo, o tipo de resposta que o computador fornece, embora útil, representa apenas uma pequena fração das críticas e orientações necessárias para que se atinja o domínio completo da arte de programar computadores. Em primeiro lugar, o interpretador não confere os resultados produzidos por um programa com os resultados esperados, ou seja, nada diz sobre a correção do programa. Em segundo lugar, nada é dito sobre a qualidade da solução obtida. Se para o primeiro tópico, já se pode contar com ferramentas verificadoras de resultados, é importante enfatizar que o segundo constitui-se em um grande desafio para os professores e monitores de cursos introdutórios. Sabe-se que quantidade e variedade de soluções que em geral precisam ser analisadas dependem da capacidade de gerá-las por parte dos alunos e da natureza da "matéria-prima" utilizada. A análise e a avaliação das soluções apresentadas são de grande importância no processo de amadurecimento do estudante, contribuindo para que o mesmo se acostume com o questionamento sobre o produto obtido, e que por certo, carece de um modelo adequado.

Normalmente, cursos introdutórios são caracterizados pela ausência de análise adequada das soluções apresentadas - um verdadeiro perigo à formação de um profissional competente. Em geral, o estudante fica muito satisfeito com o fato de seu programa funcionar. O ideal é que a solução do estudante seja criticada incisivamente, mostrando de forma clara o tipo de inadequação cometida quanto à qualidade do programa apresentado. Conseqüentemente, a tarefa de avaliação torna-se bastante árdua, já que, para o desenvolvimento satisfatório do estudante é preciso que ele resolva uma quantidade significativa de problemas de programação.

O trabalho aqui relatado é parte de um esforço multi-institucional que busca definir caminhos para atenuar a tarefa de avaliação. Buscamos os elementos necessários para a construção de assistentes capazes de analisar as soluções apresentadas, classificá-las e a elas responder com um comentário adequado à compreensão do estudante, conforme apresentado em [Castro et al, 2002]. A partir da uma análise de uma coleção de problemas resolvidos por alunos foi feita uma classificação das variações de solução, o que nos permitiu a elaboração de uma estrutura conceitual. Na seqüência formalizamos este conhecimento com o objetivo de ter uma descrição do conhecimento a ser usado pelos agentes de software que se pretende construir. Foram utilizados no presente trabalho, programas descritos em programação funcional, produzidos por estudantes em seu primeiro curso de programação. A linguagem de programação utilizada é Haskell [Bird & Wadler, 1988], uma linguagem funcional.

2. Identificando Esquemas de Programa

Para identificar as classes de soluções apresentadas pelos estudantes, definiu-se como etapa inicial desse trabalho, a análise das soluções apresentadas nas resoluções de suas

listas de exercícios das disciplinas introdutórias de programação funcional, dos cursos de Engenharia da Computação e Ciência da Computação, de duas Instituições de Ensino Superior brasileiras ao longo de três semestres letivos, totalizando seis turmas de calouros de uma instituição e duas da outra.

Quando do início dos trabalhos, observou-se que não havia necessidade de se verificar todas as listas de exercícios (o que seria um esforço muito grande, sem maiores ganhos), pois seria perfeitamente possível identificar padrões de programas, inicialmente, apenas com alguns exercícios. Face os objetivos de nosso trabalho, apenas as soluções que produzem resultados corretos para o plano de teste foram consideradas. Para cada exercício selecionado, os professores responsáveis construíram uma solução, aqui denominada de solução base, tendo em conta o nível de maturidade dos estudantes e os princípios que haviam sido apresentados e discutidos nas aulas. Procedeu-se então o estudo e, para cada exercício selecionado foi feito um levantamento dos vários tipos de solução.

Do conjunto de soluções disponíveis foi realizada uma triagem, para eliminar as soluções redundantes, produzindo-se com isso, uma lista de soluções distintas. Tomando-se como referência a solução base, cada solução proposta pelos estudantes foi analisada, visando encontrar elementos que possibilitem estabelecer diferenças entre elas, permitindo que, com isso, fosse construída uma árvore das variações encontradas para cada exercício. Em alguns casos, a solução de um estudante possibilitou a identificação de uma nova possibilidade de resolução de problemas (ou uma variação mais elegante), e quando isto ocorreu, as soluções iniciais foram redefinidas e as soluções daquele item do trabalho, reclassificadas.

2.1. Subconjunto de Haskell considerado

Para este estudo consideramos apenas os temas abordados na fase inicial do curso, cobrindo o trecho da linguagem apresentado a seguir através de sua gramática.

<definição> ::= <nome> {parâmetro} = <expressão> [<definição local>]
<expressão> ::= <expressão simples> | <expressão condicional>
<expressão condicional> ::= **if** <expressão predicativa> **then** <expressão> **else** <expressão>
<definição local> ::= **where** <definição>

3. Esquemas Conceituais – natureza do conhecimento

Como resultado das sucessivas etapas de classificação e organização das soluções apresentadas pelos estudantes, foi possível classificar as variações de soluções em ***variações sintáticas*** e ***variações cognitivas***. As variações sintáticas referem-se às diferenças entre as formas de construção obtidas a partir das possibilidades permitidas pela linguagem. Quanto às cognitivas, pode-se dizer que estão relacionadas com os diferentes modelos mentais que os estudantes constroem para elaborar a solução de um mesmo problema. Foram percebidas também variações semânticas, que não são analisadas neste trabalho por não serem o foco principal desta pesquisa. A seguir, apresenta-se uma discussão mais detalhada dos dois primeiros tipos de variação citados

e uma sugestão de uso da figura de ***evidenciação*** como estratégia para obtenção de variação semântica.

3.1. Variações Sintáticas

É importante que se identifique às variações sintáticas de uma solução para que possamos identificá-las como soluções equivalentes à solução base. As principais variações sintáticas identificadas foram:

i. Uso desnecessário do <if...then...else>, quando a expressão era do tipo boolean. Por exemplo:

   f x = if x>0 then True else False quando bastaria usar f x = x>0;

ii. Uso desnecessário de parêntesis. Por exemplo:

   if x>0 then (if ...) else ... onde bastaria usar if x>0 then if ... else ...;

iii. Uso diferenciado do <if...then...else> no que diz respeito ao domínio de interpretação do problema.

   Exemplo 1: if x>0 then 1 else 0 ou if x<0 then 0 else 1.

   Exemplo 2: if x>0 then <exp1> else <exp2> ou if not x>0 then <exp2> else <exp1>;

iv. Uso da comutatividade dos operadores. Exemplo: <exp1>&&<exp2> é similar a <exp2>&&<exp1>

3.2. Variações Cognitivas

Entende-se por variações cognitivas aquelas que dizem respeito diretamente à forma de estruturação da solução pelo estudante. Neste ponto do aprendizado, espera-se que os conceitos presentes na solução estejam diretamente relacionados com os conceitos encontrados na descrição do problema ou que possam ser diretamente depreendidos desta.

De um ponto de vista mais geral, podemos identificar a existência de diferentes metáforas para um dado problema. Cada metáfora é determinada por um conjunto de conceitos. Estes conceitos podem ser registrados pelo estudante, em diferentes níveis de abstração, produzindo, com isso, uma variedade de soluções. Entendemos que a percepção de abstrações no grau adequado produzirá soluções de melhor qualidade. Portanto, é de extrema relevância que o estudante seja incentivado e recomendado a usá-las.

A fim de fornecer um aporte teórico para a pesquisa relatada neste trabalho, baseamo-nos na taxonomia de Bloom [Bloom, 1956], no que concerne a níveis de competências cognitivas (níveis de abstração) relacionados a algumas demonstrações de habilidades típicas para cada nível. Essas competências são:

a. conhecimento
b. compreensão
c. aplicação
d. análise

e. síntese

f. avaliação

Ao se estudar pela primeira vez sobre um tópico de história, por exemplo, onde apenas se consegue visualizar datas, eventos e lugares, vislumbrando uma visão bem abrangente da situação estudada, está-se na fase mais básica do desenvolvimento cognitivo (a). Já quando se vai aprofundando mais nas leituras, começa-se obter o entendimento das questões relativas ao tema, bem como seu significado, podendo até predizer conseqüências (b). Com o decorrer de outras leituras, já se consegue utilizar as informações obtidas e aplicá-las em um outro contexto. Nesta fase, atingiu-se a etapa (c). As duas etapas seguintes ocorrem quase que simultaneamente. Na primeira (d), reconhece-se os significados escondidos ("lê-se as entrelinhas"), identificando-se novos componentes; na segunda (e) generaliza-se a partir de fatos conhecidos e já se pode tirar conclusões a respeito do tema. Na última fase do desenvolvimento cognitivo (f) o sujeito compara e discerne as idéias, faz escolhas baseadas em argumentos raciocinados, reconhece a subjetividade dos fatos e consegue valorar as evidências.

Neste trabalho, partindo da classificação das habilidades cognitivas citadas anteriormente, identificamos alguns de percepção de um determinado conceito, evidenciados em programas de alunos dos cursos sujeitos do experimento:

i. ausência de registro

ii. destaque

iii. nomeação

iv. parametrização

v. generalização

A situação em que o estudante não deixa qualquer vestígio de que tenha se apercebido do conceito é chamado de ausência de registro (i), relacionado à competência (*a*). Já o primeiro indício de que a abstração foi levada em conta é quando o estudante explicita através do uso de mecanismos de parentetização (ii), relacionado à competência (*b*). Ao dar um nome, o estudante deixa claro, principalmente pela escolha deste nome, que o conceito abstrato foi capturado (iii), relacionado à competência (*c*), já que essa ação propicia a oportunidade de utilização de um mesmo conceito mais de uma vez. Com a parametrização (iv) atinge-se o nível mais elevado da percepção de conceitos e de suas interações, relacionado às competências (*d* e *e*). Com a criação de funções paramétricas auxiliares o mesmo conceito pode ser utilizado com mais flexibilidade em diferentes situações. A generalização (v) pode trazer benefícios para a clareza do programa, mas não se constitui, necessariamente, em um artifício que o torna melhor que os demais, por produzir quando mal utilizada, códigos confusos. Esta última é, por isso, uma característica relacionada à competência (*f*).

Algumas vezes, a solução apresentada pelo estudante não deixa clara como os principais elementos relevantes à solução foram capturados, com respeito ao domínio do problema. Outras vezes identifica uma total despreocupação com a eficiência e a clareza da solução. Estas situações podem ser mais bem entendidas através da seqüência de soluções para um determinado problema, mostrada na Tabela 1.

Observando as soluções da Tabela 1, nota-se que as soluções 3 e 4 ressaltam o fato de que *a* e *b* possuem a mesma propriedade. Isto não é mostrado com clareza nas soluções 0, 1 e 2. A solução 0 apresenta uma solução simplista do problema, não capturando abstrações relevantes do domínio do mesmo. Essa solução não exibe de forma clara a idéia de abstração de intervalo. Como a solução desejada deve guardar informações sobre o problema, e tendo em vista que isto não ocorre na solução 0, ela é considerada como inflexível. A solução 1 apenas destaca a abstração de intervalo, o que mostra indícios de que o estudante percebeu a relevância do conceito. Em 2 observa-se não apenas o destaque da abstração, como também sua nomeação. Em 3 identifica-se uma parametrização (tal solução facilita sua reutilização sempre que necessário). E, finalmente, em 4 encontra-se uma generalização do problema apresentado incluindo os registros de abstração identificados nas soluções anteriores. A solução 4 inclui ainda o tratamento geral de intervalos através da função descritora de intervalos (*pertence*).

3.2.1. Generalização ×Reutilização

A solução 4 da Tabela 1, descrita na subseção anterior, oferece algumas vantagens sobre as demais. Estas vantagens podem ser consideradas tanto do ponto de vista de entendimento do leitor (legibilidade), quanto do ponto de vista de modificação do problema (reutilização). Ilustraremos esta situação através de propostas de modificação no problema original. Para cada uma apresentamos a nova solução, evidenciando a naturalidade com que o *script* é ajustado para a nova requisição. Entende-se por script um conjunto de definições de funções necessária à resolução de um determinado problema.

i. Transformação do problema 1 para um outro intervalo fechado (Tabela 2).

Enunciado: Avalie se, dados dois números *a* e *b*, ambos estão no intervalo [2..8].

Tabela 1 - Variações Cognitivas de um Problema

Problema 1 – Avalie se, dados dois números, *a* e *b*, ambos estão no intervalo [0..6]	
Solução 0	*Ausência de Registros*
Prob1 a b = a>=0 && a<=6 && b>=0 && b<=6	
Solução 1	*Destaque*
Prob1 a b = (a>=0 && a<=6) && (b>=0 && b<=6)	
Solução 2	*Nomeação*
prob a b = prob1 && prob2 where prob1 = a>=0 && a<=6 prob2 = b>=0 && b<=6	
Solução 3	*Parametrização*
prob a b = prob1 a && prob1 b where prob1 x = x>=0 && x<=6	

Solução 4	*Generalização*
pertence x ni nf = x>=ni && x<=nf -- prob a b = prob1 a && prob1 b where prob1 x = pertence x 0 6	

Tabela 2 – Mudança de Valores

Solução 4ª
pertence x ni nf = x>=ni && x<=nf -- prob a b = prob1 a && prob1 b where prob1 x = pertence x 2 8

ii. Transformação do problema 1 para um um intervalo aberto (Tabela 3).

Enunciado: Avalie se, dados dois números, *a* e *b*, ambos estão no intervalo (0..6).

Tabela 3 - Intervalo Aberto

Solução 4b
pertence x ni nf = x>ni && x<nf -- prob a b = prob1 a && prob1 b where prob1 x = pertence x 0 6

iii. Transformação do problema 1 para parametrizar a função com três elementos (Tabela 4).

Enunciado: Avalie se, dados três números *a*, *b* e *c*, os três estão no intervalo [0..6].

Tabela 4 - Função com Três Elementos

Solução 4c
pertence x ni nf = x>=ni && x<=nf -- prob a b c = prob1 a && prob1 b && prob1 c where prob1 x = pertence x 0 6

3.2.2. Generalizações Inconvenientes

Embora a generalização seja interessante para o caso apresentado na seção anterior, nem sempre seu uso pode ser considerado apropriado. O caso a seguir ilustra uma dessas situações.

Problema 2: Dados 3 números, *a*, *b* e *c*, determine a média aritmética dos extremos.

É fácil perceber que o problema apresenta, em seu enunciado, a citação de duas constantes: (i) *a quantidade de números a serem considerados (n)* e, (ii) a *quantidade de números sobre os quais determinaremos a média aritmética*.

Neste caso, podemos destacar os seguintes aspectos:

- O valor de *n* interfere na quantidade de construções do tipo *if-then-else* a serem usadas;
- Nesse estágio, outros recursos da linguagem não são dominados pelo estudante, o que inviabiliza a construção de soluções mais genéricas;
- Conseqüentemente, a generalização é cabível com relação ao domínio do problema, mas não se encaixa no domínio da solução pela falta de recursos;
- É preciso, neste caso, restringir-se aos dados do problema.

Para o domínio do problema aqui apresentado, destacam-se dois conceitos relevantes que podem ser facilmente capturados na descrição da solução: o menor número e o maior número. Há ainda, obviamente, a abstração *média aritmética*, conforme é mostrado na Tabela 5. É comum também uma solução um pouco menos refinada, com o uso da parentetização, conforme ilustra a Tabela 6.

Tabela 5 - Abstração de "menor" e "maior"

Solução 2ª
mediaDosExtremos a b c = mediaArit where mediaArit = (menor + maior) / 2.0 menor = if a<b then if a<c then a else c else if b<c then b else c maior = if a>b then if a>c then a else c else if b>c then b else c

Tabela 6 - Parentetização

Solução 2b
mediaDosExt a b c = ((if a<b then if a<c then a else c else if b<c then b else c) + (if a>b then if a>c then a else c else if b>c then b else c)) / 2.0

3.3. Evidenciação – uma estratégia para melhorar a legibilidade

Intuitivamente, a idéia de evidenciação de uma solução aparece como um tipo de variação cognitiva relevante, o que foi constatado analisando os programas dos estudantes. Antes de discutirmos alguns exemplos, é interessante afirmar o que

consideramos uma evidenciação. É o isolamento de um termo que aparece várias vezes em uma descrição, como na expressão aritmética abaixo:

$$a * b + (c + d) * b + h + (e + f - 3) * b + g$$

onde a multiplicação por *b* aparece várias vezes. Sua evidenciação nos levaria a:

$$h + g + [a + (c + d) + (e + f - 3)] * b$$

Vejamos ainda um outro exemplo, agora com um programa em Haskell, mostrado na Tabela 7.

Tabela 7 - Metade do Menor Número

Problema 3: Dados dois números, a e b, determine a metade do menor deles.	
Solução 0	*Comentário*
prob3 a b = if a<b then a / 2.0 else b / 2.0	O estudante perspicaz notará rapidamente que esta solução pode ser reescrita, evidenciando a divisão por 2.0, produzindo a solução 1.
Solução 1	*Comentário*
prob3 a b = (if a<b then a else b) / 2.0	Esta operação destaca a abstração "menor de dois números", através da expressão (if a<b then a else b). Se ainda, essa operação é nomeada, chega-se à solução seguinte.
Solução 2	*Comentário*
prob3 a b = menor / 2.0 where menor = if a<b then a else b	Se quisermos refiná-la ainda mais podemos generalizá-la, como mostra a solução 3.
Solução 3	*Comentário*
menor x y = if x<y then x else y -- prob3 a b = menor a b / 2.0	Com esta solução, foi atingido um alto grau de refinamento.

Como podemos observar, partimos da intenção em isolar a divisão por 2.0 e acabamos por atingir uma generalização, através da valorização do conceito *menor*, que anteriormente não era explicitado. A conclusão que chegamos neste caso é que o tipo de solução que atingimos inevitavelmente também se manifestaria se tivéssemos buscado a identificação de conceitos relevantes no domínio do problema. Por outro lado, acreditamos também que quando produzimos uma solução, podemos usar a operação de evidenciação como uma fonte de busca de conceitos ocultos. Sendo assim, decidimos considerá-la como uma heurística na árdua tarefa de construir soluções abstratamente convenientes.

Percebemos ainda que há fortes indícios de que a evidenciação seja uma ferramenta de grande utilidade quando estivermos trabalhando no domínio de soluções, buscando aquelas que possuam um melhor desempenho computacional. Por exemplo, no caso da expressão que apresentamos no início desta discussão, observa-se uma redução de duas operações de multiplicação.

Ilustremos um pouco mais a situação, revisitando o problema 2, mostrado no Quadro 1.

Quadro 1 - Revisitando o Problema 2

Problema 2 – Dados 3 números, *a*, *b* e *c*, determinar a média aritmética dos extremos
Solução 2a
mediae a b c = if (a>=b) && (a>=c) then caso1 else if (b>=a)&&(b>=c) then caso2 else caso 3 where caso1 = (a + (if b>=c then c else b)) / 2.0 caso2 = (b + (if a>=c then c else a)) / 2.0 caso3 = (c + (if a>=b then b else a)) / 2.0
Solução 2b
mediae a b c = (if (a>=b)&&(a>=c) then caso1 else if(b>=a)&&(b>=c) then caso2 else caso3) / 2.0 where caso1 = (a + (if b>=c then c else b)) caso2 = (b + (if a>=c then c else a)) caso3 = (c + (if a>=b then b else a))

4. Formalização dos Esquemas Conceituais (modelagem do conhecimento)

Para a representação do conhecimento será utilizada a lógica de primeira ordem, por ser uma linguagem capaz de expressar univocamente os pressupostos sugeridos pela pesquisa.

4.1. Estruturação

Cada variante metafórica de um problema possui uma lista de conceitos, os quais temos expectativas que o estudante abstraia. Para descrever uma solução, usa-se *scripts*, os que podem ser modelados pela utilização de componentes sintáticos. Cada solução possui um identificador. Escolhemos modelar a descrição de uma solução utilizando dois símbolos relacionais: *CompCognitivo* e *CompSintatico*. Ambos binários, representando a associação entre uma solução e um componente cognitivo ou sintático, respectivamente. Cada componente sintático e cada componente cognitivo serão descritos por uma relação associando um identificador com uma descrição. A primeira denominamos de *elemSintatico* e a segunda de *elemCognitivo*. As diversas variações, sintática ou cognitiva, são modeladas pelos símbolos predicativos *TipoVarSintatica* e *TipoVarCognitiva*, respectivamente.

CompCognitivo(<solução>,<componente cognitivo>)
CompSintatico(<solução>,<componente sintático>)
ElemSintatico(<identificador>,<descrição>)
ElemCognitivo(<identificador>,<descrição>)
TipoVarSintatica(<tipo de variação sintática>)
TipoVarCognitiva(<tipo de variação cognitiva>)

Solução(<solução>)

As soluções estão ligadas entre si por uma ou mais variações sintáticas ou por uma ou mais variações cognitivas. No primeiro caso modelamos usando o símbolo predicativo *VarSintatica* e no segundo *VarCognitiva*. Em qualquer uma das duas relações envolvemos as duas soluções, o componente e o tipo de variação.

VarSintatica(<solução 1>, <solução 2>, <componente sintático>, <tipo de variação>)
VarCognitiva(<solução 1>, <solução 2>, <componente cognitivo>, <tipo de variação>)

4.2. Axiomatização

Nesta etapa do trabalho identificamos alguns elementos para a descrição mais apurada do conhecimento envolvido na descrição de soluções e suas interações. A seguir, listamos o resultado de uma primeira exploração das interrelações e suas particularidades. Entendemos que há muito ainda para ser identificado, o que esperamos atingir em projetos futuros.

1) a relação de variação sintática é irreflexiva;
2) a relação de variação cognitiva é irreflexiva;
3) a relação de variação sintática é anti-simétrica;
4) a relação de variação cognitiva é anti-simétrica;
5) Cada solução não pode possuir mais de uma origem de derivação sintática;
6) Cada solução não pode possuir mais de uma origem de derivação cognitiva;
7) Entre duas soluções, as variações cognitivas em um dado conceito, são únicas;
8) Entre duas soluções, as variações sintáticas em um dado componente, são únicas;
9) Toda solução, exceto a solução abstrata (solução 0), possui uma solução da qual deriva sintática ou semanticamente;
10) As soluções abstratas só possuem variantes cognitivas metafóricas;
11) As soluções abstratas não possuem variantes sintáticas.

4.2.1. Representação Lógica dos Axiomas

ax1: $\forall$s1, s2, c1, t1 VarSintatica(s1,s2,c1,t1) $\Rightarrow$ $\neq$(s1,s2)
ax2: $\forall$s1, s2, c1, t1 VarCognitiva(s1,s2,c1,t1) $\Rightarrow$ $\neq$(s1,s2)
ax3: $\forall$s1, s2, c1, t1 VarSintatica(s1,s2,c1,t1) $\Rightarrow$ $\sim\exists$c2,t2 VarSintatica(s2,s1,c2,t2)
ax4: $\forall$s1, s2, c1, t1 VarCognitiva(s1,s2,c1,t1) $\Rightarrow$ $\sim\exists$c2,t2 VarCognitiva(s2,s1,c2,t2)
ax5: $\forall$s1, s2, s3, c1, c2, t1, t2 VarSintatica(s1,s2,c1,t1) $\wedge$ VarSintatica(s3,s2,c2,t2) $\Rightarrow$ =(s1,s3)
ax6: $\forall$s1, s2, s3, c1, c2, t1, t2 VarCognitiva(s1,s2,c1,t1) $\wedge$ VarCognitiva(s3,s2,c2,t2) $\Rightarrow$ =(s1,s3)
ax7: $\forall$s1, s2, s3, c1, c2, t1, t2 VarSintatica(s1,s2,c1,t1) $\wedge$ VarSintatica(s3,s2,c2,t2) $\Rightarrow$ =(c1,c2) $\wedge$ =(t1,t2)
ax8: $\forall$s1, s2, s3, c1, c2, t1, t2 VarCognitiva(s1,s2,c1,t1) $\wedge$ VarCognitiva(s3,s2,c2,t2) $\Rightarrow$ =(c1,c2) $\wedge$ =(t1,t2)
ax9: $\forall$s solução(s) $\wedge$ $\neq$(s,solução-0) $\Rightarrow\exists$s1,c1,t1 VarSintatica(s1,s,c1,t1)
$\vee$ $\exists$s2,c2,t2 VarCognitiva(s2,s,c2,t2)
ax10: $\forall$s,s1,c,t solução(s) $\wedge$ =(s,solução-0) $\wedge$ VarCognitiva(s,s1,c,t)
$\Rightarrow$ =(t,metafórica)
ax11: $\forall$s,s1,c,t solução(s) $\wedge$ =(s,solução-0) $\Rightarrow$ $\neg\exists$s1, c, t VarSintatica(s,s1,c,t)

4.2.2. Aplicação dos Axiomas

Os axiomas apresentados na seção anterior foram definidos após a observação do desempenho de quatro turmas de alunos calouros, totalizando dois semestres letivos. Após esses resultados iniciais, aplicamos o conhecimento inferido às outras turmas de que trata esta pesquisa, incrementando os procedimentos experimentais com testes de lógica, questionários e entrevistas.

Os resultados obtidos até o presente momento confirmam a axiomatização descrita e confirmam a validade dessa classificação para os outros conceitos abordados nesses cursos introdutórios, como o uso de listas e os paradigmas aplicativo e recursivo.

5. Conclusões

Neste trabalho buscamos identificar componentes importantes das soluções apresentadas por estudantes de disciplinas introdutórias de programação, dentro de uma abordagem funcional, visando compará-las com soluções consideradas mais adequadas do ponto de vista da legibilidade.

A identificação e a classificação das modalidades de variações cognitivas possibilitaram a modelagem das interligações entre soluções, definindo uma linguagem de primeira ordem. Ademais, vislumbra-se uma metodologia para o ensino de programação, voltada para o grau de legibilidade e reutilização de uma solução. Nesta direção, podemos mostrar ao aluno como comparar soluções visando melhorar as qualidades acima citadas.

Além da extensão dos axiomas apresentados na Seção 4.2 e a preparação de material para cursos de programação apoiados nas idéias relatadas, a construção de ferramentas baseadas em conhecimento modelado conforme ilustrado aqui, é um desdobramento natural desse trabalho.

Referências

Bird, R.S., Wadler, Ph., Introduction to Functional Programming, Prentice Hall, ISBN 0-13-484197-2, 1988, New York.

Bloom, B. S. (ed.) *Taxonomy of Educational Objectives: the classification of educational goals. Handbook I – cognitive domain*. 1956, New York.

Castro, T., Castro Jr, A. N., Menezes, C. S., Boeres, M. C. S., Rauber, M. C. P. V. Utilizando Programação Funcional em Disciplinas Introdutórias de Computação In: XXII Congresso da Sociedade Brasileira de Computação / X Workshop sobre Educação em Computação, 2002, Florianópolis.

Giegerich, R., Hinze R., Kurtz, S. Straight to the Heart of Computer Science via Functional Programming. Workshop on Functional and Declarative Programming in Education. 1999. Paris, FR.

Joosten, S., Van-den-Berg, K., Van-der-Hoeven, G. Teaching Functional Programming to First-Year Students. Journal of Functional Programming. Vol.3, N.1. 1993.

Mckeown, J., Farrell, T. Why We Need to Develop Success in Introductory Programming Courses. CCSC - Central Plains Conference. 1999. Maryville, MO.

Proposta para Desenvolvimento de Metodologia de Ensino e de Ferramental de Acessibilidade para a Qualificação Profissional de Deficientes Visuais e Motores

Cláudia Medronho Naumann
Universidade Federal do Rio de Janeiro, Núcleo de Computação Eletrônica
Bloco C do CCMN, Cidade Universitária, Caixa Postal 2324, CEP 20001-970,
Rio de Janeiro, Brasil
naumann@nce.ufrj.br

e

Sergio Guedes de Souza
Universidade Federal do Rio de Janeiro, Núcleo de Computação Eletrônica,
Bloco C do CCMN, Cidade Universitária, Caixa Postal 2324, CEP 20001-970,
Rio de Janeiro, Brasil,
guedes@nce.ufrj.br

Abstract

According to 2000 Cense, there are approximately 24.5 million people in Brazil who have some kind of deficiency, where the visually and mobility impaired are the majority. The government actions are mainly taken toward to basic education, so not many advanced education programs are destined to impaired people. Although some specialized institutions usually offer several technical courses, they are considered basic-level, and don't prepare them to get a more qualified job. On the other hand, the number of corporations that offer jobs to them is reduced. Our institution NCE/UFRJ has been developing accessibility tools since 1993, which provides access to information through, but not only, the Internet. Based on this context, NCE/UFRJ has created the Projeto Habilitar, which the objective are to providing intermediate to high-level courses in both administrative and technical areas for impaired people, to develop a teaching methodology and to develop or upgrade accessibility tools to be used in future courses and by other institutions as well. The strategy applied to start this project was to deploy a pilot course. The course chosen is called Cisco Networking Academy Program, implemented by Cisco Systems Inc., the biggest corporation in networking technology of the world.
Keywords: Deficiency, Digital Inclusion, Special Teaching, Technical Qualifying, Development of Technology for Impaired People.

Resumo

De acordo com o Censo 2000, existem aproximadamente 24 milhões de habitantes no Brasil que possuem algum tipo de deficiência, nos quais os portadores de deficiência visual e motora são a maioria. As ações governamentais são geralmente direcionadas à educação básica, de modo que poucos programas de educação são voltados aos deficientes. Embora algumas instituições especializadas ofereçam alguns cursos técnicos, eles são considerados de nível básico e, portanto não os preparam para obter um emprego que exija qualificações. Por outro lado, o número de empresas que oferecem empregos a eles é bastante reduzido. O NCE/UFRJ vem desenvolvendo ferramentas de acessibilidade desde 1993, provendo acesso à informação através, mas não somente, da Internet. Baseado nesse contexto, o NCE/UFRJ criou o Projeto Habilitar, cujos objetivos são oferecer cursos de nível médio e superior nas áreas técnica e administrativa para portadores de deficiência, desenvolver uma metodologia de ensino e desenvolver ou atualizar ferramentas de acessibilidade a serem utilizadas em cursos futuros e por outras instituições. A estratégia adotada para iniciar esse projeto foi implementar um curso piloto. O curso escolhido é intitulado Cisco Networking Academy Program, desenvolvido pela Cisco Systems Inc, uma das maiores empresas de tecnologia de redes do mundo.
Palavras-chaves: Deficiência, Inclusão Digital, Ensino Especial, Capacitação Profissional, Desenvolvimento de tecnologia para portadores de deficiência.

1. Cenário do Deficiente no Brasil

No Brasil, segundo os dados do Censo/2000 [1] realizado pelo Instituto Brasileiro de Geografia e Estatística (IBGE) e resumidos na Tabela 1, cerca de 24,5 milhões de brasileiros são portadores de deficiência, o equivalente a 14,5% da população [8]. De acordo com a Tabela 2 [1], podemos observar que a deficiência visual possui uma alta ocorrência, cerca de 9,8% da população seguido pela deficiência motora (4,6%).

População Brasileira (2000)	**169799170**	**100%**
Total de deficientes	**24537984**	**14.5%**

Tabela 1: Total de portadores de deficiência no Brasil

O avanço social da maior parte dos deficientes no Brasil é limitado pela baixa renda, acesso muito limitado à cultura formal e políticas públicas inadequadas ao desenvolvimento social. Esse imenso contingente encontra-se, em sua maioria, alijado do mercado de trabalho por falta de capacitação profissional [9]. Um dos fatores que levam a este cenário é a ausência de uma política de incentivo a programas de capacitação de deficientes por parte das Instituições Públicas e Privadas.

A questão da responsabilidade social empresarial é um tema atual de grande importância em todo mundo, e em especial no Brasil [10]. Entretanto, enquanto nos países do Primeiro Mundo existem diversos fundos de apoio formados por ações de empresas socialmente responsáveis, no Brasil a situação ainda é incipiente. No país, essas ações ganharam forte impulso na década de 90, através de Organizações Não Governamentais (ONG's), institutos de pesquisa e empresas sensibilizadas para a questão. De fato, a enorme desigualdade social existente dá à responsabilidade social empresarial uma relevância ainda maior. A sociedade de uma forma geral espera que as empresas cumpram um novo papel no processo de desenvolvimento: que sejam agentes de uma nova cultura e que sejam promotores de uma mudança social, com vistas a construir uma sociedade melhor.

Deficiência mental permanente	**2848684**	**1.7%**
Deficiência visual	**16573937**	**9.8%**
Incapaz de enxergar	159824	0.1%
Grande dificuldade permanente de enxergar	2398472	1.4%
Alguma dificuldade permanente de enxergar	14015641	8.3%
Deficiência motora	**7879601**	**4.6%**
Incapaz de caminhar ou subir escada	588201	0.3%
Grande dificuldade permanente de caminhar ou subir escada	1799917	1.1%
Alguma dificuldade permanente de caminhar ou subir escada	5491482	3.2%
Deficiência auditiva	**5750809**	**3.4%**
Incapaz de ouvir	176067	0.1%
Grande dificuldade permanente de ouvir	860889	0.5%
Alguma dificuldade permanente de ouvir	4713854	2.8%
Tetraplegia, paraplegia ou hemiplegia permanente	**955287**	**0.6%**

Tabela 2: Total de portadores de deficiência por tipologia

No tocante à questão de trabalho da pessoa portadora de deficiência (PPD), a maioria das empresas ainda efetua contratos de emprego obrigadas pela legislação vigente [2]. Porém, mais que contratar apenas porque a lei manda, a inserção dos deficientes na empresa se torna também atrativa do ponto de vista empresarial e, acima de tudo, converge para os anseios da sociedade [12]. Uma justificativa econômica para empregar PPDs é que eles podem ser caracterizados como um recurso produtivo inexplorado, que se instrumentado tecnologicamente de forma conveniente, pode alcançar um ótimo desempenho. Pesquisas demonstram que trabalhadores portadores de deficiência geralmente atingem altos níveis de desempenho no trabalho e possuem maiores índices de freqüência e devoção [13]. Empregar PPDs também

beneficia a imagem da empresa, que passa a ser reconhecida como "boa empregadora", além de promover um processo de humanização que se reflete diretamente nos seus empregados e clientes. Por outro lado, empresas tidas como discriminadoras tendem a serem consideradas como não socialmente responsáveis, tornando-se a "última opção" para busca de empregos, tanto para não portadores de deficiência como para PPDs, além da imagem negativa perante aos clientes.
Diversas empresas que possuem em seus quadros PPDs reconhecem que empregá-los melhorou o ambiente de negócios, posto que permite a elas espelhar com maior precisão a estrutura social das comunidades onde desejam se inserir. A inserção do PPD no ambiente de trabalho, de uma forma geral, requer somente pequenas alterações em escalas de trabalho e algumas modificações no ambiente físico e/ou nos equipamentos.
No entanto, a absorção de deficientes pelas empresas, no Brasil, ainda é aquém do exigido por lei. Uma das alegações para tal é que os cargos oferecidos não podem comportar portadores de deficiência devido à falta de qualificação. Por outro lado, as instituições tradicionais de apoio ao deficiente, não oferecem condições reais de formação para ascensão profissional, uma vez que se preocupam, em sua maioria, em disponibilizar apenas cursos básicos à comunidade de deficientes. Mesmo quando são oferecidos cursos de capacitação de alto nível, as ferramentas de acessibilidade ou não se aplicam, ou são mal utilizadas.
Desta forma, o cenário brasileiro é composto por deficientes com formação profissional inadequada e em desigualdade de condições para conquista de emprego. Existe, contudo, a necessidade da implantação de programas de capacitação profissional que propiciem ao portador de deficiência a oportunidade de ingressar no mercado de trabalho, não somente para obter uma fonte de renda, mas também para ascender profissionalmente permitindo a ele o resgate da sua cidadania.

2. Motivação para a criação do Projeto Habilitar

O Núcleo de Computação Eletrônica da Universidade Federal do Rio de Janeiro (NCE/UFRJ), vem realizando, desde 1993, atividades de ensino e pesquisa voltadas a portadores de deficiência física visual e motora [3]. Esse trabalho é focado basicamente no desenvolvimento de ferramental que possibilite ao deficiente físico o acesso à informação, e dessa forma promover uma perspectiva de reintegração social. Dentre as diversas ferramentas desenvolvidas se destacam o DOSVOX [4], software que utiliza síntese de voz para traduzir ao deficiente visual o que é mostrado na tela do computador, e o MOTRIX [5], software que provê acesso ao computador através de comandos de voz fornecidos pelo deficiente motor. Essas ferramentas têm sido largamente utilizadas em âmbito nacional, uma vez que elas são distribuídas gratuitamente por todo o país através da Internet. Uma versão em espanhol desses softwares está em desenvolvimento, de modo que eles sejam disponibilizados para a América Latina em um futuro próximo.
A distribuição gratuita dessas ferramentas vem ao encontro das necessidades dos portadores de deficiência visual e motora que possuem baixo poder aquisitivo, uma vez que as ferramentas comerciais existentes no mercado têm um custo elevado.
O Projeto Habilitar [6], criado pelo NCE/UFRJ em março de 2003, possui como objetivo a capacitação profissional do deficiente, não somente através da oferta de cursos nas áreas técnica e administrativa, mas também acompanhar o seu encaminhamento ao mercado de trabalho, realizando convênios com instituições públicas ou privadas. O primeiro passo em direção ao mercado de trabalho é a obtenção de estágio, sendo um passo seguinte a contratação do deficiente.

3. Fase inicial do Projeto Habilitar

Antes de iniciar qualquer atividade no Projeto, foi estabelecido que ele iria, em uma primeira instância, abranger somente a comunidade de PPDs visuais e motores, uma vez que a nossa instituição já possui uma larga experiência com esses dois tipos de deficiência. Foram também definidos que os cursos a serem oferecidos a essa comunidade estariam dispostos em duas categorias: técnica e administrativa. Na área técnica selecionamos os cursos de técnico em redes de computadores, programador JAVA e Webmaster, enquanto na área administrativa optamos pelos cursos de Auxiliar de Biblioteca e Assistente de Administração.
A princípio, esses cursos não teriam nenhuma dificuldade em serem oferecidos, uma vez que todos eles já possuem ementas padrão, e em nosso caso, como pertencemos a uma universidade, não teríamos problema algum em conseguir instrutores qualificados. No entanto, surge uma questão: seriam esses cursos aplicados a PPDs da mesma forma que são aplicados a indivíduos não portadores de deficiência? Logicamente teria o mesmo conteúdo, mas não podemos dizer o mesmo no caso da metodologia e do ferramental de ensino. Ao pensarmos nesse ponto, concluímos que não possuíamos instrutores totalmente qualificados para tal tarefa.
Dessa forma, decidimos iniciar o Projeto com um único curso, um curso piloto, um laboratório propriamente dito, que tivesse como principal objetivo desenvolver metodologia de ensino e utilizar/desenvolver ferramentas de acessibilidade que pudessem ser utilizadas em cursos futuros, e disseminadas a outras instituições de ensino.

Nossa opção recaiu em iniciar o Projeto através de um curso técnico em redes de computadores posto que o NCE/UFRJ tem uma equipe de profissionais de Redes de Computadores com larga experiência em ensino, pesquisa e prestação de serviços, e também por ser a Coordenação de Qualidade e de Ensino de um programa de capacitação em redes chamado Cisco Networking Academy Program da Cisco Systems Inc [3]. Outro forte incentivo foi o apoio que a Cisco do Brasil Ltda e a Cisco Systems Inc decidiram prestar ao Projeto, destacando-se ofertas de estágio aos alunos do curso e a participação e premiação dos melhores alunos na instalação da infra-estrutura de redes no evento internacional da Cisco Systems Inc, o Networkers 2003.

O curso está dividido em quatro módulos, cada um com carga horária de 80 horas, com aulas ministradas duas vezes por semana com 4 horas de duração. Ele consiste de uma parte teórica e uma parte prática que deve ser realizada em um laboratório devidamente montado. Todo o material é acessível via Web. A avaliação do aluno é feita através de provas de múltipla escolha, que devem ser realizadas online, e dependendo do módulo, provas práticas em laboratório. Em termos educacionais, o material do curso não foi projetado para atender a PPDs.

Uma turma piloto foi constituída por 12 alunos, sendo 6 deficientes visuais (4 com nenhuma visão e 2 com visão sub-normal) e 6 deficientes motores graves (com diferentes níveis de tetraplegia), de forma a capacitá-los em uma profissão de alto reconhecimento no mercado e possibilitar o desenvolvimento de uma metodologia de ensino, bem como verificar as necessidades de desenvolvimento ou de adaptação de ferramentas de apoio à atividade de ensino. Durante o andamento do curso são efetuadas análises e estudos relacionados ao uso e adequação destas ferramentas e metodologias aplicadas a PPDs e a instrutores envolvidos na atividade de ensino.

Toda a experiência adquirida nesta turma piloto servirá como balizador na implantação de outros cursos e na disseminação a outras instituições que queiram participar do Projeto.

4. Desenvolvimento de Metodologia e de Ferramental

Para dar início ao curso foi necessário selecionar uma equipe que pudesse trabalhar diretamente com os alunos da turma piloto. A equipe teria que ser composta em uma primeira instância de dois instrutores, um monitor; uma pedagoga e um psicólogo. O ideal seria que toda a equipe tivesse experiência prévia no trato de deficientes, e como isso não foi possível, tivemos que construir toda uma metodologia de ensino apoiada somente na experiência técnica dos profissionais envolvidos.

Apesar de termos em nosso quadro um número considerável de profissionais aptos a ministrar esse curso, foi muito difícil conseguirmos adesões, uma vez que a maioria se mostrou temerosa em ter algum tipo de contato com deficientes. No entanto, conseguimos um número reduzido de instrutores que se engajou no Projeto. Com os monitores também foi difícil, conseguindo apenas dois. O monitor é uma peça chave em sala de aula, uma vez que ele tem como principal tarefa auxiliar o instrutor nas aulas práticas, bem como interagir diretamente com os alunos no que concerne à adaptação e utilização das ferramentas de acessibilidade. Uma pedagoga foi convocada a trabalhar conosco de forma a orientar os instrutores em termos de didática e de metodologia de ensino.

O psicólogo teria como tarefa prestar apoio emocional tanto a instrutores e monitores, uma vez que eles não possuem nenhuma experiência com deficientes, quanto a alunos, pois muitos dos deficientes necessitam de apoio psicólogo não somente para vencer a deficiência, como também para encarar a sociedade de perto, já que muitos deles viveram por muito tempo afastados do convívio social. Infelizmente até o presente momento não pudemos contar com o apoio de um profissional de psicologia.

Como a turma é composta por alunos com dois tipos de deficiência, foi necessário fazer uma avaliação prévia de como distribuí-los de forma eficiente em sala de aula. Decidimos dividí-los em grupos de dois alunos, cada um composto por um deficiente visual e um deficiente motor. Dois foram os fatores que contribuíram para tal decisão; o primeiro foi acreditarmos que a interação entre indivíduos proporcionaria o entendimento mútuo das dificuldades, de modo que cada um pudesse ajudar a suprir a deficiência do outro. O outro fator foi devido aos deficientes visuais terem maior nível de escolaridade, e isso poderia contribuir positivamente no ensino dos deficientes motores.

Em suma, o deficiente motor contribuiria com a sua visão, e o deficiente visual com a sua mobilidade e com o seu conhecimento. Até o presente momento essa organização em sala de aula tem tido resultados excelentes. O convívio social dentro e fora da sala de aula tem contribuído para a troca de experiências e resolução prévia de dúvidas proporcionando um alto grau de interação entre eles.

Sendo a turma composta por diferentes tipos de escolaridade, foi decidido iniciar o curso com um pequeno treinamento em plataforma Windows. Esse treinamento serviu para analisar se as ferramentas já desenvolvidas iriam atender às necessidades dos deficientes, e o resultado obtido foi positivo. De fato as ferramentas proporcionaram aos deficientes o acesso ao computador, e, por conseguinte, à Internet.

Para dar início e continuidade ao curso é necessário que duas metodologias de ensino sejam desenvolvidas concomitantemente, cada uma voltada a um determinado tipo de deficiência. A integração das duas se dará ao longo do curso. Cabe ao instrutor procurar a melhor forma de explanação, seja para o deficiente visual, seja para o deficiente motor.

Uma característica importante comum às duas deficiências é a incapacidade dos alunos em fazer anotações de aula. É importante, que, de tempos em tempos, sejam realizadas atividades práticas em sala de aula, seja através de execução de aplicativos ou de exercícios em laboratório, de forma a aliar a teoria com a prática para melhor fixar o conhecimento, e tornar aulas a princípio teóricas menos tediosas.

No caso do deficiente motor, ele deve ter a sua capacidade visual explorada ao máximo por parte do instrutor e do monitor. Mesmo alguns deles possuindo alguns movimentos dos membros superiores, estes não podem ser considerados movimentos coordenados, o que impossibilita tarefas relacionadas à montagem e instalação de equipamentos. No entanto, cabe ao instrutor capacitá-lo a ser capaz, no âmbito do curso de redes, a gerenciar essas tarefas e ser capaz de identificar problemas de conexão e de comunicação utilizando a sua visão e as ferramentas de acessibilidade.

Um procedimento normal adotado por instrutores em turmas convencionais é a de se escrever um conjunto de informações no quadro-negro sem falar nada. Esse tipo de prática jamais pode ser adotada em uma sala de aula com deficientes visuais. Tudo que for apresentado em sala de aula na forma de escrita ou de objeto deve ser descrita com o máximo de detalhes possível, de forma que o deficiente visual possa entender o que está sendo apresentado.

Outra prática comum é a de se perguntar se o aluno está vendo o que está escrito no quadro, ou no caso de um curso de redes, se ele está vendo um determinado equipamento. Deve-se evitar fazer tal pergunta a um deficiente visual, mas se isso acontecer, o instrutor deve estar preparado para a reação do aluno, e para não se sentir mal com esse tipo de situação. O instrutor e o monitor devem tratar de explorar ao máximo a capacidade tátil do deficiente visual. O manuseio de cabos, conectores e equipamentos de redes como placas de redes, roteadores e comutadores devem ser largamente explorados.

Foi observado durante algumas aulas que a impossibilidade dos deficientes visuais de terem acesso às figuras do material estava comprometendo a compreensão do conteúdo do curso, e por esse motivo decidiu-se adotar formas de se apresentar as figuras a eles.

Caso a figura tivesse um caráter mais descritivo, ela era convertida para um formato texto e posteriormente disponibilizada aos alunos via Internet. Uma outra opção, que tem se mostrado bastante eficaz, foi a representação de figuras em maquetes ou em baixo-relevo. Esta abordagem tornou possível aos deficientes visuais, por exemplo, reconhecer e entender as diversas topologias de redes existentes. Ela se apresentou como uma opção em relação à representação pelo método Braille, que implicaria na existência de um equipamento de custo elevado [3].

Na confecção de maquetes optamos pela utilização de materiais de custo baixo e de fácil obtenção e que obtivessem o resultado desejado. Foram utilizados folhas de papel, cordas finas, botões de roupa, cola, papelão e palitos. O tipo de material adotado é apenas uma sugestão, podendo ser utilizados ainda palitos de sorvete para substituir os barbantes e tampinhas de garrafas no lugar dos botões. Não foi utilizada nenhuma técnica específica na construção das maquetes, uma vez que a elaboração depende muito da criatividade de cada instrutor/monitor e do material disponível. Deve-se, no entanto, se estabelecer uma padronização para a aplicação em cursos regulares e de larga escala, de modo a evitar a ocorrência de problemas de compreensão.

Em nossa turma piloto, por exemplo, para representar uma topologia física de rede, optamos por utilizar uma corda fina para representar o meio físico entre os equipamentos e botões de distintos tamanhos para representar os diferentes equipamentos. Para a construção das maquetes imprimimos as figuras das topologias de rede em um tamanho que facilitasse a colagem dos diversos materiais por sobre o papel, e que pudesse prover uma boa "visão" espacial através do tato. Em seguida cortamos as cordas finas de acordo com o tamanho dos enlaces, que variavam entre retas e circunferências. Colamos então as cordas finas nos locais referentes aos meios físicos, de forma a evitar sobras, já que nos extremos de cada corda fina existem os nós representados por botões. Deve-se ter o cuidado de não haver diferença de relevo entre as cordas finas e os nós [3]. Papelões de diversos tamanhos foram colados no papel para diferenciar os diversos tipos de equipamentos como PC´s, switches e hubs. Conseguimos ainda o apoio de uma instituição para cegos para imprimir, em Braille, as legendas a serem adicionadas as maquetes.

Tanto as figuras convertidas para texto como as convertidas para maquetes foram submetidas a uma avaliação prévia dos deficientes visuais para validar a compreensão tátil da representação. Este procedimento é fundamental para evitar que haja interpretações dúbias. O resultado foi surpreendente, pois à medida que as maquetes eram apresentadas aos alunos, o reconhecimento por parte deles era imediato. Essa forma de representação foi, e continua sendo, de extrema importância na consolidação do conhecimento teórico apresentado em sala de aula, e não somente para os deficientes visuais, mas também para os deficientes motores, que as utilizaram como material complementar de estudo.

Por ser um curso não projetado para deficientes, não era possível o ingresso de um indivíduo portador de deficiência até o presente momento. Houve, portanto, a necessidade não só de adaptar o material do curso, de forma a promover uma melhor compreensão do conteúdo, mas também de desenvolver uma série de ferramentas que pudessem proporcionar o acesso do deficiente ao curso.

Para a leitura do material, via Web, foi utilizado o DOSVOX para os deficientes visuais, e o MOTRIX para os deficientes motores. O MONIT32, um leitor de tela integrante do conjunto de aplicativos do DOSVOX, foi utilizado para a realização das provas teóricas on-line pelos deficientes visuais. Com a ajuda de um instrutor ou monitor não

portador de deficiência, o deficiente visual faz uma marcação prévia dos lugares onde estará posicionado o enunciado da questão e as opções a serem selecionadas. Importante que a prova seja configurada de modo a apresentar uma questão por página, de modo que a marcação possa ser feita corretamente. Esta operação é uma das poucas que necessitam de intervenção de um monitor em virtude do MONIT32 não estar totalmente adaptado as provas on-line da Cisco Systems Inc. Embora tal intervenção ocorra, os deficientes visuais fazem as provas sozinhos. O MOTRIX é utilizado pelos deficientes motores na realização das provas. No caso do MOTRIX, a prova pode apresentar mais de uma questão por página.

Como o curso contém uma ampla variedade de laboratórios práticos, foram desenvolvidas ferramentas específicas para o deficiente visual, embora também possam ser utilizadas para os deficientes motores em uma versão modificada, para auxiliar na percepção do que acontece na rede, bem como na configuração de equipamentos de rede, como por exemplo, roteadores e switches.

Para capturar pacotes na rede, foi desenvolvido o VOXDUMP, que na verdade é uma interface amigável, utilizando a síntese de voz, acoplada ao software de domínio público para captura de pacotes na rede, o WINDUMP. Como nesse software existem vários parâmetros e combinações de captura, optamos por implementar, nessa primeira versão, as opções mais comuns de captura e apresentá-las de forma mais simplificada aos deficientes visuais, de forma que eles não tivessem que se preocupar com ordem nem formato de parâmetros.

Por ser um curso de formação de técnicos de redes, grande parte desse curso está voltada à configuração de roteadores e switches. Para que os deficientes visuais pudessem configurar esses equipamentos tanto local como remotamente, foi utilizado um aplicativo denominado TELNETVOX, integrante do conjunto de aplicativos do DOSVOX, sendo necessárias algumas adaptações para atender a esse fim. A configuração de roteadores utilizando o TELNETVOX foi bem sucedida, enquanto que os testes com switches ainda estão em andamento.

Durante o curso são mostrados cálculos que devem ser feitos utilizando-se lápis e papel, como é o caso, por exemplo, de conversão de números de base decimal para binária e de divisão de um endereço de rede em sub-redes. Esse tipo de atividade deve ser apresentado com muita cautela, uma vez que os deficientes visuais e motores não possuem tal facilidade. Portanto, para facilitar o aprendizado, foi desenvolvida uma calculadora de conversão de base em duas versões, uma para cada tipo de deficiência, chamada CALCBASE, uma vez que a calculadora existente no Windows não atende às necessidades dos deficientes. No caso dos deficientes visuais, a voz é utilizada para informar o resultado das operações de conversão de base.

Para a divisão de um endereço de redes em sub-redes desenvolveu-se o VOXCALC, onde através da voz, os deficientes visuais aprendem como subdividir os endereços de rede Internet. Essa ferramenta também teve uma versão operada por comando de voz destinada aos deficientes motores. As duas ferramentas têm se mostrado bastante eficazes na consolidação de conhecimento dos alunos, embora a sua utilização seja apenas para auxiliar no aprendizado do respectivo cálculo, e não como substituto ao raciocínio.

5. Dificuldades Encontradas

Alguns problemas foram encontrados durante a fase inicial do Projeto que tiveram que ser contornados. A acessibilidade foi o primeiro problema que foi solucionado através da total remodelação do acesso ao NCE/UFRJ em função das cadeiras de rodas. Rampas de acesso foram construídas, banheiros adaptados e acessos laterais reformados.

O laboratório para a realização do curso foi cedido em sua totalidade pelo NCE/UFRJ, contendo um PC para cada aluno, kits multimídia para a utilização dos softwares DOSVOX e MOTRIX, todo o ferramental desenvolvido/adaptado ao longo do curso e todos os equipamentos de redes como roteadores, switches e hubs. O NCE/UFRJ ainda financia o coffee-break dos alunos.

A locomoção dos deficientes motores das suas residências até o NCE/UFRJ era um obstáculo a ser transposto para a realização do curso. Como o Projeto não possui recursos para subsidiar tal transporte, uma parceria com uma empresa de ônibus público se fez necessária. Um ônibus dessa empresa foi totalmente adaptado para o transporte do deficiente motor, assegurando a sua participação no curso.

O suporte prestado pelos instrutores ao Projeto, sem haver nenhum tipo de financiamento externo tem sido de suma importância para o seu andamento, uma vez que temos encontrado dificuldades em conseguir novas adesões. De fato, nenhum dos instrutores possuía experiência prévia no ensino de deficientes, e conseqüentemente muitos não sabiam como se portar profissionalmente perante aos alunos. Dentre os instrutores que foram convidados a participar do Projeto, houve um certo número que se recusou a participar, enquanto os que aceitaram foram acometidos por descontrole emocional, que incluiu dores de cabeça, dificuldade para dormir, entre outros sintomas.

A falta de um acompanhamento psicológico para os alunos e professores, nessa turma piloto, foi um problema de certa forma minimizado pela dedicação dos instrutores, monitores e dos próprios alunos. No entanto, para a continuação dos próximos módulos, mais estressantes, esta falta precisa ser corrigida.

6. Conclusão

Atualmente a turma piloto se encontra finalizando o módulo 2, e temos como previsão de término dos 4 módulos o mês de setembro de 2004. O desempenho acadêmico dessa turma tem sido acima do esperado para uma experiência piloto, como podemos observar nas Tabelas 1 e 2 [7], denotando que os métodos e ferramentas desenvolvidas até o presente momento são adequadas e eficazes. Do ponto de vista qualitativo, as notas obtidas nas avaliações têm sido excelentes quando comparadas a uma turma de não deficientes no mesmo período e utilizando os mesmos instrutores.

Tabela 1 – Média das notas obtidas nos exames parciais e finais do módulo I.

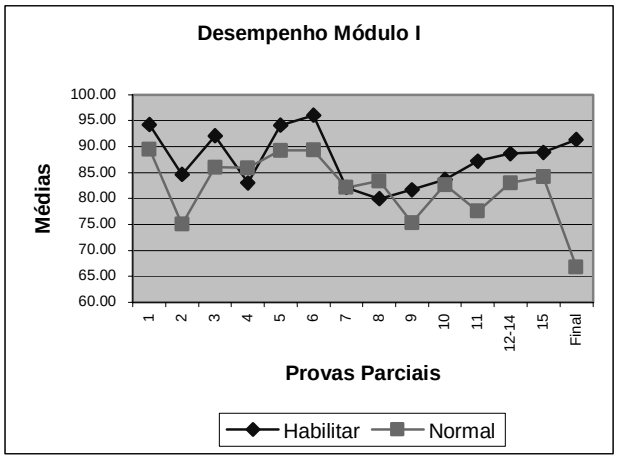
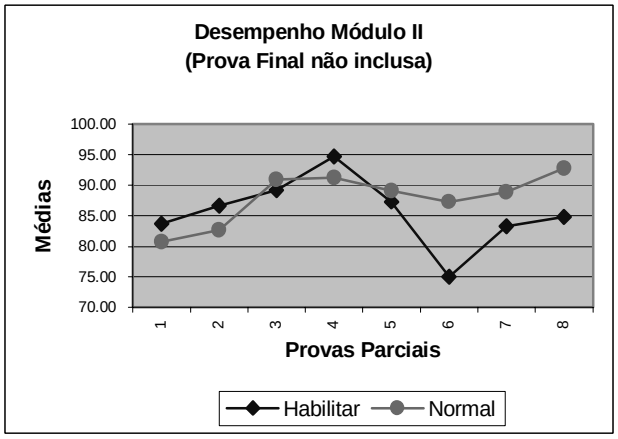


Tabela 2 – Média das notas obtidas nos exames parciais do módulo II.



Analisando o comportamento da turma piloto pelos gráficos constantes nas tabelas apresentadas, notamos uma queda de desempenho da turma Normal, no módulo I, na prova final. Este resultado foi decorrente de 3 alunos não haverem realizado a prova por motivos diversos.
No módulo II, observa-se uma queda de desempenho da turma Habilitar, nas provas parciais 6, 7 e 8. Isto decorreu da ausência de um aluno, por motivos de saúde, durante a realização das provas.
Excetuando-se essas discrepâncias nos dados, podemos perceber que o desempenho da turma Habilitar é superior a turma Normal no módulo I e bastante similar no módulo II.

7. Agradecimentos

Ao NCE/UFRJ, que nos forneceu pronto apoio no primeiro instante, efetuando alterações em seu espaço físico, tal como a construção de uma rampa de acesso, de modo a prover suporte apropriado a todos os indivíduos portadores de deficiência motora.
A Cisco do Brasil Ltda, Rio de Janeiro e São Paulo, pelo pronto atendimento a todas as nossas reivindicações.

Ao Sr. Marco Cobb, da Cisco Systems Inc, Costa Rica, pelo seu incentivo e atuação em favor do Projeto Habilitar, no Brasil e nos Estados Unidos.
A Sra. Silvia Wygand, da Cisco do Brasil Ltda, Rio de Janeiro, por estar sempre atenta, em qualquer cliente, para uma oportunidade de inserção dos portadores de deficiência, em especial os do Projeto Habilitar.
Aos professores da turma piloto, pela coragem de participar de uma experiência um tanto ambiciosa.
Ao Sr. Sérgio Alberto F. da Rocha, Coordenador do NCE/UFRJ no biênio 2002-2004, pela percepção social deste projeto.
Aos nossos alunos da turma piloto pela vontade, caráter, senso comunitário e coragem de se comprometerem a enfrentar um desafio e se deparar com problemas que há muito haviam deixado de lado.

8. Referências

[1] IBGE, Censo Demográfico 2000 – Tabulação Avançada, Instituto Brasileiro de Geografia e Estatística, disponível em http://www.ibge.gov.br/censo/default.php, Brasil, 2000.
[2] Ministério da Justiça, "Lei 7853", Decreto n. 3298, Diário Oficial da União, Brasil, Dezembro, 1999.
[3] Naumann, C. M., Souza, S. G., Borges, J. A., "Projeto Habilitar: Deficiente Físico X Mercado de Trabalho", Anais do III Congreso Iberoamericano Iberdiscap 2004, San José, Costa Rica, Março, 2004.
[4] Borges, J. ª, "Le Projet DOSVOX – Comment changer la vie de milliers d'aveugles brésiliens", Disabled Magazine, UNESCO, 1998.
[5] CNN Espanhol, Entrevista ao Programa Adelantos, Segmento Horizontes, Disponível em http://intervox.nce.ufrj.br/motrix, Janeiro, 2003.
[6] Naumann, C. M., Souza, S.G., "Projeto Habilitar: Capacitação Profissional Para Pessoas Portadoras de Deficiência", Documentação do Projeto NCE/UFRJ, Rio de Janeiro, RJ, Brasil, Agosto, 2003.
[7] Cisco Networking Academy Program, "Gradebook de alunos", Academia NCE/UFRJ, Rio de Janeiro, RJ, Brasil, 2003.
[8] Néri, M. C., Retrato da Deficiência, Revista Conjuntura Econômica, Fundação Getúlio Vargas, julho, 2003, pág. 42-45, Rio de Janeiro, RJ, Brasil.
[9] Mattar, M. E., Eficiência com as diferenças, RITS, La Insígnia, maio, 2003, http://www.lainsignia.org/2003/mayo/soc_024.htm.
[10] Néri, M. C., Retratos das pessoas com deficiências ao longo dos tempos, disponível online em http://www.saci.org.br, Seção ARTIGOS, junho, 2003.
[12] Sassaki, R., Construindo uma sociedade para todos, WVA Editora e Distribuidora Ltda. – 3ª ed. – 2002.
[13] Manpower Inc., Working with Disabilities, Technical Report, Manpower Report 436, abril, 2003.
[14] Naumann, C. M., Souza, S. G., "Habilitar: um Projeto de Capacitação Profissional para Deficientes Visuais e Motores Através do Desenvolvimento de Metodologia de Ensino e de Ferramentas de Acessibilidade", Simpósio Iberoamericano de Educação, Cibernética e Informática, SIECI 2004, Orlando, Flórida, Julho, 2004.

Ensino de compiladores apoiado por um ambiente virtual de aprendizagem

Silvana Rossy de Brito[1,2], Aleksandra do Socorro da Silva[1,2,3], Eloi Luis Favero[1], Maria da Penha de Andrade Abi Harb[1]
[1] Universidade Federal do Pará (UFPA)
Programa de Pós-Graduação em Engenharia Elétrica(PPGEE)
Belém, Pará, Brasil, 66.075-110
{srossy, aleka, favero, mpenha}@ufpa.br

[2] Instituto de Estudos Superiores da Amazônia (IESAM)
Núcleo de Tecnologias Interativas de Aprendizagem (NUTEIA)
Belém, Pará, Brasil, 66055-260

[3] Universidade da Amazônia (UNAMA)
Belém, Pará, Brasil, 66060-902

e

Orivaldo de Lira Tavares
Universidade Federal do Espírito Santo (UFES)
Centro Tecnológico (CT)
Vitória – ES, Brasil, 29060-900
tavares@inf.ufes.br

Abstract

Among the pedagogical possibilities used to assist learning of Compilers, the project oriented learning favouring the integration between disciplines and the development of abilities. This article presents the compilers study in context the approach of projects, relating some of the difficulties faced for teachers and presenting the authors experience with the use of a virtual environment of learning.
Keywords: compilers, learning virtual environment, project oriented learning, graduation.

Resumo

Dentre as possibilidades pedagógicas utilizadas para auxiliar a aprendizagem de Compiladores, a pedagogia de projetos se destaca por favorecer a integração entre disciplinas e o desenvolvimento das habilidades envolvidas no processo de desenvolvimento de um projeto de engenharia de software. Este artigo aborda o estudo de compiladores segundo a abordagem de projetos, relatando algumas das dificuldades enfrentadas pelos professores e apresentando a experiência dos autores com a utilização de um ambiente virtual de aprendizagem.
Palavras-chave: Compiladores, ambiente virtual de aprendizagem, aprendizagem baseada em projetos, graduação.

1. INTRODUÇÃO

Nos cursos de graduação em computação, o ensino de diversas disciplinas vem sendo valorizado com a abordagem de projetos. Bons resultados têm sido alcançados com o desenvolvimento de projetos em Engenharia de Software, Sistemas Operacionais, Linguagens de Programação e Compiladores. A pedagogia de projetos destaca-se por valorizar a integração entre disciplinas, além de favorecer a cooperação e a colaboração entre estudantes, valorizando as habilidades de trabalho em grupo, de coordenação e gerência de projetos. Essa abordagem é particularmente adequada em disciplinas onde o desenvolvimento do projeto aparece de forma faseada, como é o caso do projeto de um compilador. Para apoiar esse processo de construção, acreditamos ser imprescindível a utilização adequada de tecnologias que apóiem a cooperação e a coordenação, características, hoje, consideradas essenciais em ambientes virtuais de aprendizagem (AVAs), e que, atualmente, é tratada com a utilização dos recursos da *web*[1]. Com o uso adequado dessas tecnologias, expandem-se as possibilidades pedagógicas uma vez que é possível favorecer o reuso de conhecimento, o compartilhamento de informações, a colaboração e a cooperação.

No contexto do ensino de compiladores, o desenvolvimento dos projetos é feito freqüentemente em equipe, motivo pelo qual a cooperação está quase sempre associada ao desenvolvimento de projetos de compiladores. O problema proposto (que é a construção do compilador) pode ser aprimorado pelos próprios estudantes e a partir de então, percebe-se o início de um processo, no qual todos os envolvidos passam a contribuir e a interagir, necessitando coordenar atividades presenciais, divisão de tarefas, reutilização de trabalho e documentação tanto do compilador quanto (como freqüentemente lhes é solicitado) do processo de construção do compilador. E é nesse cenário que uso de AVAs pode acrescentar facilidades tanto do ponto de vista do professor, quanto do estudante.

O estudo de compiladores, apesar de freqüentemente envolver projeto e construção de um compilador, tem como objetiva não necessariamente a construção do compilador em si, mas estudar e entender os princípios, técnicas e ferramentas usadas. Esse conhecimento é útil em inúmeras aplicações, principalmente hoje que a maior parte da informação disponível está na Internet em forma de linguagem natural e de objetos semi estruturados, como por exemplo em XML[2]. Assim, o planejamento da disciplina deve prever que o curso se consolide nessas reflexões e não apenas na construção do artefato em si.

Este artigo apresenta o relato da experiência com a utilização de um ambiente interativo de aprendizagem na disciplina de compiladores, discutindo suas principais dificuldades, vantagens e necessidades. Está estruturado da seguinte forma: a seção 2 apresenta os objetivos do estudo de compiladores em cursos de computação; a seção 3 discute a abordagem de projetos; a seção 4 relata algumas das dificuldades enfrentadas pelos professores; a seção 5 apresenta a experiência com a utilização de um ambiente virtual de aprendizagem para apoiar o ensino de compiladores utilizando a abordagem de projetos e na seção 6, as conclusões deste trabalho.

2. OBJETIVOS DA DISCIPLINA

Compiladores são ferramentas de tradução entre linguagens, mantendo a semântica original, tais como: ambientes para linguagens de programação (compiladores, interpretadores, *debuggers*, *profilers*, etc), ambientes para o processamento de linguagens naturais (verificadores *orto-sintáticos* e tradutores), ferramentas para a compatibilização entre dispositivos de hardware (*device-drivers*, emuladores, *cross-compilers*, etc.), dentre outras [21].

O estudo de Compiladores deve abordar: (i) a estrutura de um compilador; (ii) a análise de programas-fonte, com o estudo dos métodos mais importantes de análise léxica e sintática, semântica, de organização das tabelas de símbolos e gerenciamento de erros; (iii) as ferramentas para a geração automática dos componentes de um compilador; (iv) máquinas abstratas e otimização de código intermediário; (v) ambientes de tempo de execução; (vi) síntese de programas-objeto, compreendendo esquemas de tradução dirigida por sintaxe, geração de código de máquina e otimização de código [21]. Espera-se que, ao final da disciplina de compiladores, os estudantes tenham a percepção concreta de que ferramentas fundamentais como métodos formais, engenharia de software, estruturas de dados e algoritmos colaboram para resolver um problema complexo. É importante que os estudantes aprendam como e por que são construídos os compiladores, além de como construí-los [5].

A tabela 1 ilustra a abrangência da disciplina de compiladores, considerando duas principais dimensões: das atividades (análise, síntese, síntese & análise) e dos níveis lingüísticos (léxico, sintático e semântico); para cada célula desta tabela temos conceitos e técnicas diferentes, com a complexidade aumentando da esquerda para a direita e de cima para baixo. Na coluna da direita temos as 4 principais disciplinas envolvidas: LF= Linguagens Formais & Autômatos; TC= Teoria da Computação; ED = estrutura de dados; OC= Organização de Computadores (além do tema TDS= Tradução dirigida por sintaxe).

---

[1] *Web* é uma abreviatura da *world wide web* (www).

[2] A sigla XML corresponde a Linguagem de Marcação Expansível (do inglês: e*Xtensible Markup Language*").

Tabela 1. Abrangência da disciplina de Compiladores

Análise Léxica	Síntese	Análise x Síntese	Disciplinas
		Erros-léxicos	LF; TC
		Tabela-símbolos	ED
Sintática		Erros-sintáticos	LF; TC
		Tabela-símbolos	ED
Semântica	TDS	Erros-semânticos	LF; ED
	TDS: Geração de código	Código de máquina: Geração & Otimização	LF; ED; OC

Assim, para atender os objetivos da disciplina e a necessidade de consolidar os conhecimentos de Teoria de Linguagens Formais e Autômatos e Teoria da Computação, a abordagem escolhida freqüentemente envolve o projeto de um compilador. É fundamental, que ao final do projeto de um compilador, o estudante desenvolva as habilidades necessárias para justificar a escolha das ferramentas, ambientes, paradigmas e linguagens utilizadas [21]. Muitos dos conceitos vistos apenas do ponto de vista teórico (como modularidade, manutenibilidade, portabilidade e custos de software) podem ser oportunsticamente analisados durante todo o desenvolvimento do projeto de um compilador.

3. APRENDIZAGEM DE COMPILADORES ATRAVÉS DE PROJETOS

Disciplinas como compiladores ou sistemas operacionais incluem freqüentemente um projeto no qual os estudantes implementam (no todo ou em parte) um pequeno exemplo do sistema em estudo. Assim, é comum o estudo de compiladores segundo a abordagem de projetos nos cursos de computação. Segundo Baldwin [5], esses projetos encorajam a aprendizagem, forçando os estudantes a entender profundamente o assunto, por meio de seus próprios exemplos. O desenvolvimento do projeto de um compilador ocorre de forma faseada, por meio da construção dos diversos componentes do compilador [9].

Com a abordagem de projetos, é possível favorecer a consolidação, de forma integrada, dos conteúdos estudados em Teoria da Computação, Engenharia de Software, Teoria de Linguagens Formais e Autômatos. Do contrário, o aprendizado dessas disciplinas, quando desvinculado da prática, tende a ser enfadonho, cansativo e pouco produtivo [12]. Uma metodologia de ensino integrada é essencial pois é na disciplina de compiladores onde acontece a consolidação entre os conceitos estudados e a prática da construção. Infelizmente, poucos professores da área de computação estão familiarizados com teorias educacionais e poucos especialistas em educação tentam aplicar teorias educacionais na educação em informática [7], embora propostas construtivistas como [6, 8, 14] tentem resumir a teoria para professores de computação, fazendo com que cada vez mais, professores de computação mencionem explicitamente o construtivismo como a base teórica para propostas pedagógicas, para a utilização de software educacional e ambientes virtuais de aprendizagem.

Na concepção da pedagogia de projetos, os projetos se constituem em planos de trabalho e conjunto de atividades que podem tornar o processo de aprendizagem mais dinâmico, significativo e interessante para o aprendiz. Para o estudo de compiladores, essa abordagem freqüentemente envolve um projeto significativo onde os estudantes escrevem um compilador para uma linguagem de programação restrita. Segundo Aiken [3], o projeto de um compilador freqüentemente tem dois objetivos distintos: a) apoiar o aprendizado sobre o projeto de uma linguagem e sobre a implementação de um compilador para essa linguagem; b), fornecer para os estudantes a experiência de construir um projeto de software significativo. No contexto de um curso de graduação em computação, um projeto de compilador pode ser a tarefa de construção de software mais complexa desempenhada pelos estudantes. Essa tarefa envolve a realização de pesquisas, a investigação e especificação de requisitos do software, o registro de dados, a formulação de hipóteses, a análise, aplicação e avaliação do artefato construído.

Para o sucesso da abordagem de projetos, o envolvimento dos aprendizes, a responsabilidade e a autonomia são fundamentais. Os aprendizes são co-responsáveis pelo trabalho e pelas escolhas ao longo do desenvolvimento do projeto [25]. Em geral, essas escolhas são realizadas em equipe, motivo pelo qual a cooperação está também quase sempre associada ao trabalho de projetos. A construção de um compilador é uma tarefa complexa e exige a cooperação de um grupo de aprendizes para realizá-la. Essa construção (especificar, escrever, testar, avaliar e documentar o compilador) constitui-se em um problema que exige o planejamento e a execução de um conjunto de atividades para a sua resolução.

4. ASPECTOS CONSIDERADOS PELOS PROFESSORES

Nesta seção discute-se as principais dificuldades dos professores em cursos de compiladores segundo a abordagem baseada em projetos. Essas dificuldades, parte relatada pela experiência dos autores e parte coletada na literatura especializada, permitiram realizar uma classificação para os problemas enfrentados segundo diferentes aspectos:

- **Quanto ao planejamento do curso:** o projeto da linguagem é o primeiro problema enfrentado no planejamento de cursos de compiladores. Segundo Aiken [3], especificações precisas devem ser escritas para apoiar as fases de projeto, implementação, testes e documentação do compilador. Idealmente, o projeto inteiro deveria ser implementado pelos projetistas do curso, antes de seu início, uma vez que somente uma implementação completa pode garantir que o projeto seja consistente, completo e rastreável. Entretanto, a idealização completa de um projeto de compilador freqüentemente não é viável para o professor. Pressões de tempo favorecem uma especificação em "tempo real" (enquanto o curso está acontecendo). Uma vez criado um projeto (especificação e implementação), o investimento feito já representa um forte incentivo para reusá-lo diversas vezes, em diferentes turmas. É assim que muitos professores criam projetos, repetindo tarefas de outros, havendo pouca reutilização entre instituições ou até mesmo entre professores da mesma instituição. Isso não tem acontecido em outras áreas que têm projetos amplamente compartilhados, como é o caso da disciplina de Sistemas Operacionais que disponibiliza simuladores, slides de cursos [20] e sistemas operacionais pedagógicos [10]. É fácil concluir que a condição de ensino melhora substancialmente se professores e instrutores compartilham os frutos de seu trabalho mais amplamente.
- **Quanto à linguagem a ser compilada:** na comunidade de linguagens de programação há um longo e útil debate sobre quais linguagens e conceitos de linguagens são importantes e devem ser ensinados. No contexto dos cursos de compiladores, as questões freqüentemente levantadas são: *Qual a linguagem a ser compilada ? Em que linguagem os estudantes devem escrever os seus compiladores ?* A experiência com a disciplina de compiladores reforça a idéia de [3] de que essas duas perguntas não são as questões mais fundamentais para a especificação de um curso de compiladores. Para atender aos objetivos da disciplina de compiladores, as preocupações que antecedem essas questões são: *o projeto está bem especificado? O projeto de compilador é possível de ser construído, documentado e avaliado?*. Ou seja, a preocupação está muito mais centrada na linguagem para a qual o compilador será desenvolvido do que nas linguagens e ferramentas nas quais será escrito. Assim, pode ser mais interessante inventar uma linguagem em vez de utilizar um subconjunto de uma linguagem existente, dado o fato que a linguagem inventada pode ser projetada para ser de fácil implementação em vez de ser fácil de utilizar (objetivo das linguagens reais). Além disso, os estudantes não ficam restritos às linguagens existentes, favorecendo a comparação entre linguagens e forçando os estudantes a pensarem conscientemente no significado das sintaxes das linguagens. Com a abordagem de projetos, constatamos a vantagem em se combinar o uso de gramáticas lógicas para prototipação e especificação de linguagens de programação. Assim, os estudantes podem completar a especificação, em gramáticas lógicas, e a implementação (numa linguagem convencional tal como C++, Pascal e Java) de uma mini-linguagem imperativa, o que só é possível devido ao poder de abstração das gramáticas lógicas. Por exemplo, no quadro 1, temos um fragmento da especificação de um comando condicional IF/THEN/ELSE e no quadro 2, o trecho de uma gramática lógica que implementa a especificação. Nota-se que o código executável em Prolog é até menor que o código da especificação.

Quadro 1: Especificação do comando IF/THEN/ELSE [24]

```
<command> ::= if <boolean expr> then <command sequence>1
                  else <command sequence>2 end if
Code(<command>) ← concat(Code(<boolean expr>),
[(JF,label(InhLabel(<command>)+1))],
Code(<command sequence>1),
[(J,label(InhLabel(<command>)+2))],
[(label(InhLabel(<command>)+1),LABEL)],
Code(<command sequence>2),
[(label(InhLabel(<command>)+2),LABEL)])…
```

Quadro 2: Gramática lógica [24]

```
command(Code,Temp,InhLab,SynLab) -->
      [if], { InhLab1 is InhLab+1, label(InhLab1,Lab) },
      booleanExpr(Code1,Temp),
      [then], commandSeq(Code2,Temp,InhLab1,SynLab), [end,if],
      { concat(Code1, [['JF',Lab]],Code2, [[Lab,'LABEL']], Code) }.
```

- **Quanto ao acompanhamento do projeto:** o desenvolvimento de um projeto de compilador é consumidor de tempo. Para Aiken [3], é impossível para a maioria dos estudantes implementar qualquer coisa além do que uma pequena linguagem. A escolha de uma linguagem real (qualquer linguagem que tenha um significativo número de usuários) é viável apenas se o projeto do compilador envolve apenas um subconjunto desta (ex: linguagem MINILISP, representada por um subconjunto da linguagem LISP). É preciso observar, também, que existe pouco valor educacional na implementação de facilidades adicionais. Portanto, o professor deve atentar para não especificar riquezas de interface, por exemplo. Os problemas que resultam das dificuldades de acompanhamento da agenda do projeto remetem às seguintes conseqüências: (1) o projeto pode não ser concluído (ficar incompleto), embora possa ter outros méritos; (2) os estudantes não terem tempo, no estágio final, para refletirem acerca da experiência vivida, diante dos demais compromissos do período letivo. É conveniente que o resultado de cada projeto desenvolvido seja materializado em um produto final, que pode ser apresentado para toda a turma. Assim, os artefatos de um pequeno grupo de aprendizes podem atrair o interesse de outros, favorecendo o refinamento do processo de aprendizagem.
- **Quanto às características do projeto especificado:** na especificação do projeto do compilador, um conjunto de características define as facilidades e os aspectos que devem ser abordados durante o desenvolvimento dos trabalhos: (1) Modularidade, portabilidade e reusablidade do projeto, já que os projetos normalmente são divididos em quatro fases (análise léxica, análise sintática, análise semântica e geração de código) e a dependência entre fases é um problema inerente do projeto do compilador. Por exemplo, um estudante que faz um analisador léxico pobre, pode ser penalizado indiretamente no sintático, porque não será possível testar o sintático completamente quando o léxico ainda contém erros, e assim por diante. Da mesma forma, sem um gerador de código funcionando corretamente, um estudante que escreve o analisador semântico não pode testar a interface de geração de código. Deve-se buscar pela independência entre as tarefas. Assim, se um estudante fizer um trabalho pobre em uma tarefa ele não deve estar em desvantagem nas tarefas seguintes. Idealmente, o projeto deve ser completamente modular: cada módulo (correspondente a uma fase) do projeto deve ser compilado separadamente e utilizado em qualquer outro compilador, desde que possua uma interface adequada àquele módulo [3]. O projeto também deve ser portátil entre plataformas, para garantir maior reusabilidade entre os diversos artefatos produzidos. Além disso, deve ser *novo,* evitando o desenvolvimento de uma "memória" (estudantes submetem como seu próprio trabalho, projetos de anos anteriores). Para evitar isso, pequenas mudanças na especificação do projeto podem desanimar esses estudantes.
- **Quanto à linguagem com a qual o projeto será implementado:** a escolha da linguagem na qual os estudantes escrevem seus compiladores não é trivial. É importante que essa escolha facilite a minimização de erros de codificação rotineiros que podem prejudicar o tempo de desenvolvimento e o entusiasmo pelo projeto. Uma desvantagem, nesse sentido, está na limitação de ferramentas para construção de compiladores que apóiem a programação em qualquer linguagem de programação. Essa decisão depende também das linguagens de programação utilizadas em outras disciplinas.
- **Quanto às ferramentas para construção de compiladores:** os cursos de compiladores utilizam, freqüentemente, ferramentas para construção de compiladores (variações do LEX e do YACC). Essas ferramentas são projetadas para agilizar o desenvolvimento do projeto, ocultando detalhes de implementação. Segundo Baldwin [5], os "compiladores para compiladores" apóiam cursos onde a meta é ensinar os estudantes a *escrever* compiladores. Entretanto, como essas ferramentas escondem detalhes de implementação das diversas fases do compilador, elas não auxiliam muito quando se espera que os estudantes compreendam como o compilador funciona. Nesse sentido, as ferramentas visuais, tais como simuladores[3] [17], podem auxiliar os estudantes na compreensão do funcionamento das partes geradas automaticamente [5].
- **Quanto aos conteúdos didáticos utilizados**: diversos livros de compiladores [4] [24] apresentam a especificação e a implementação de um compilador (como exemplo) para um subconjunto de uma linguagem real. Além dos exemplos existentes na literatura, outros compiladores (de outros projetos) podem ser apresentados como "estudos de caso" [1]. Esses compiladores são bem aplicados como demonstrações, mas seus módulos nem sempre são tão reutilizáveis quanto é desejável, além do que eles não compilam pequenas linguagens, como seria o caso ideal [5]. O próprio título do clássico livro de compiladores, conhecido como "o livro do dragão" (*Compiladores, princípios técnicas e ferramentas)* [2] - sugere que o objetivo final do estudo não necessariamente é a construção de um compilador mas sim estudar e compreender os princípios, as técnicas e as ferramentas usadas na construção de um compilador. Esses conhecimentos são úteis para inúmeras aplicações principalmente hoje que a maior parte da informação disponível está na Internet em forma de linguagem natural e de objetos semi estruturados (por exemplo em XML).

[3] Simuladores são usados na educação com o intuito de proporcionar aprendizagem sobre os conceitos que permeiam o sistema que está sendo simulado. Os simuladores podem ser usados para imitar o funcionamento de partes do compilador, como por exemplo, simuladores de autômatos, que podem ser utilizados para demonstrar o funcionamento do analisador léxico.

- **Quanto à colaboração entre os aprendizes**: deficiências na comunicação entre professores, estudantes e colaboradores afetam a colaboração entre os estudantes. Segundo Grégoire & Laferrière [13], um projeto será bem sucedido se a participação dos aprendizes e suas contribuições tiverem sido importantes para o grupo de maneira geral. Os artefatos são os produtos gerados ou consumidos nas diversas atividades durante a construção do compilador, podendo ser os produtos (ou subprodutos) gerados nas diversas fases de compilação.
- **Quanto à abordagem de projetos**: de um modo geral, é possível reunir alguns problemas que derivam da utilização da abordagem baseada em projetos. Esses problemas caracterizam-se como pequenas armadilhas que podem prejudicar os objetivos da disciplina. São elas: (1) os estudantes que não conseguem completar as fases iniciais podem permanecer em desvantagem ao longo do projeto; (2) concentrados no projeto, os estudantes podem refletir pouco sobre os benefícios da aprendizagem; (3) o curto espaço de tempo pode comprometer a conclusão do projeto. Vale ressaltar que até mesmo um compilador pequeno pode amedrontar estudantes que não possuem experiências anteriores com a abordagem de projetos. Uma solução para esses problemas está na modificação de uma implementação (na verdade, na implementação parcial do compilador), em vez de criar um compilador inteiro desde o início. Essa abordagem é popular em cursos de sistemas operacionais [16].

5. UTILIZANDO UM AMBIENTE VIRTUAL DE APRENDIZAGEM

Com o advento das novas Tecnologias da Informação e Comunicação (TIC) é possível criar um novo e amplo espaço de possibilidades para a Educação [19]. Como exemplo, a cooperação e a colaboração entre pessoas geograficamente distantes, vêm sendo facilitadas com o apoio da teleinformática e dos serviços da Web que suportam os ambientes virtuais de aprendizagem. Esses serviços incluem áreas para compartilhamento de arquivos (escaninhos para entrega de trabalhos; estantes para a publicação de documento, programas e para a especificação de trabalhos); fóruns de discussão; salas de bate-papo; e-mails; ferramentas para acompanhamento do projeto; etc. Esses recursos agregam vantagens tais como [23]: o reuso de conhecimento, o compartilhamento de informações e a cooperação, além de possibilitarem a interação entre pessoas com diferentes interesses e níveis de conhecimento.

A primeira experiência dos autores em utilizar um ambiente de aprendizagem baseado na Web, aconteceu em 2000, com a implantação do Laboratório Virtual de Compiladores [18]. Com a utilização contínua desse ambiente, foi possível aprimorar os projetos de compiladores, por meio do compartilhamento de projetos anteriores. Uma vez que projetos anteriores podem servir como referência aos novos projetos, os estudantes podem utilizá-los como exemplos de soluções para os novos projetos. Assim, os estudantes podem desenvolver a habilidade de comparar seus compiladores. Nesse contexto, Aiken [3] sugere também que se utilize um compilador de referência para garantir que o projeto do compilador seja possível de ser construído pelos estudantes em um período curto de tempo. O laboratório virtual de compiladores disponibilizava alguns recursos de simulação e disponibilização de material, entretanto, devido a pouca interatividade do ambiente, dificultava a manutenção e atualização, assim como limitava o uso por parte dos estudantes, a disponibilização de material de apoio. A experiência evoluiu para a utilização de um ambiente interativo de aprendizagem, o TelEduc [26], em duas turmas de compiladores no ano de 2003. Essa experiência permitiu um levantamento de novas dificuldades e possibilitou a especificação de um novo ambiente [15] com novas facilidades de compartilhamento e colaboração entre os estudantes.

O TelEduc é um ambiente virtual de aprendizagem, disponibilizado atualmente como software livre, pela UNICAMP. Utilizamos, atualmente, a versão 3.2.1 para acompanhar os projetos de compiladores, em diferentes turmas de graduação em computação e engenharia da computação. A figura 1 mostra uma tela (visão do professor) do TelEduc em um curso de compiladores, da turma de Engenharia da Computação, na Universidade Federal do Pará. O curso é presencial e é acompanhado através dos recursos disponíveis pela ferramenta (agenda, atividades, material de apoio, leituras, fóruns e murais, correio interno, formação de grupos , portfólios individuais e de grupo) conforme mostra a figura. Adicionalmente, o sistema também permite gerar relatórios de utilização das ferramentas, recurso que pode estar disponível tanto para professores, quanto para os estudantes.

Para auxiliar os estudantes a compreenderem os benefícios da aprendizagem de compiladores, os cursos foram organizados segundo a abordagem de projetos, nos quais os estudantes escreveram pequenos compiladores. Os projetos foram importantes para expor os estudantes às fases de compilação. A idéia do curso é de até 50% da carga horária com teoria, laboratório e orientação e 50% para trabalho em grupo, onde são desenvolvidos projetos de compiladores para mini-linguagens, considerando uma disciplina com 60 a 90 horas aula. Isto só é possível com ferramentas de alto nível que permitem a especificação e prototipação das gramáticas das mini-linguagens, nos três níveis lingüísticos (léxico, sintático e semântico/geração-de-código). Pela nossa experiência, se fornecemos a especificação de uma gramática lógica, por exemplo, em uma disciplina de até 60 horas/aula, as equipes podem desenvolver seus projetos cobrindo as três principais fases (análise léxica, sintática e semântica/geração de código). Nesses casos, são pré-requisitos uma disciplina de Linguagens Formais, uma disciplina ou curso rápido de Programação em Prolog (20 horas no mínimo) e conhecimentos de programação e estruturas de dados. Utilizamos os textos de [24] e [11] que apresentam compiladores completos especificados e implementados em Prolog, com o uso de gramáticas lógicas. Em Prolog uma mini-linguagem pode ser descrita em algumas centenas de linhas,

tipicamente de 100 a 300 linhas. Estes compiladores foram o ponto de partida do desenvolvimento dos projetos. Adicionalmente, mostramos como podemos codificar uma gramática lógica em uma linguagem convencional como C++, Pascal, Java. Basicamente adotamos uma abordagem descendente recursiva, onde cada regra gramatical é codificada como uma função *booleana*. Os atributos da regra são passados como parâmetros. Com isso o estudante é guiado na codificação pela especificação executável em Prolog.

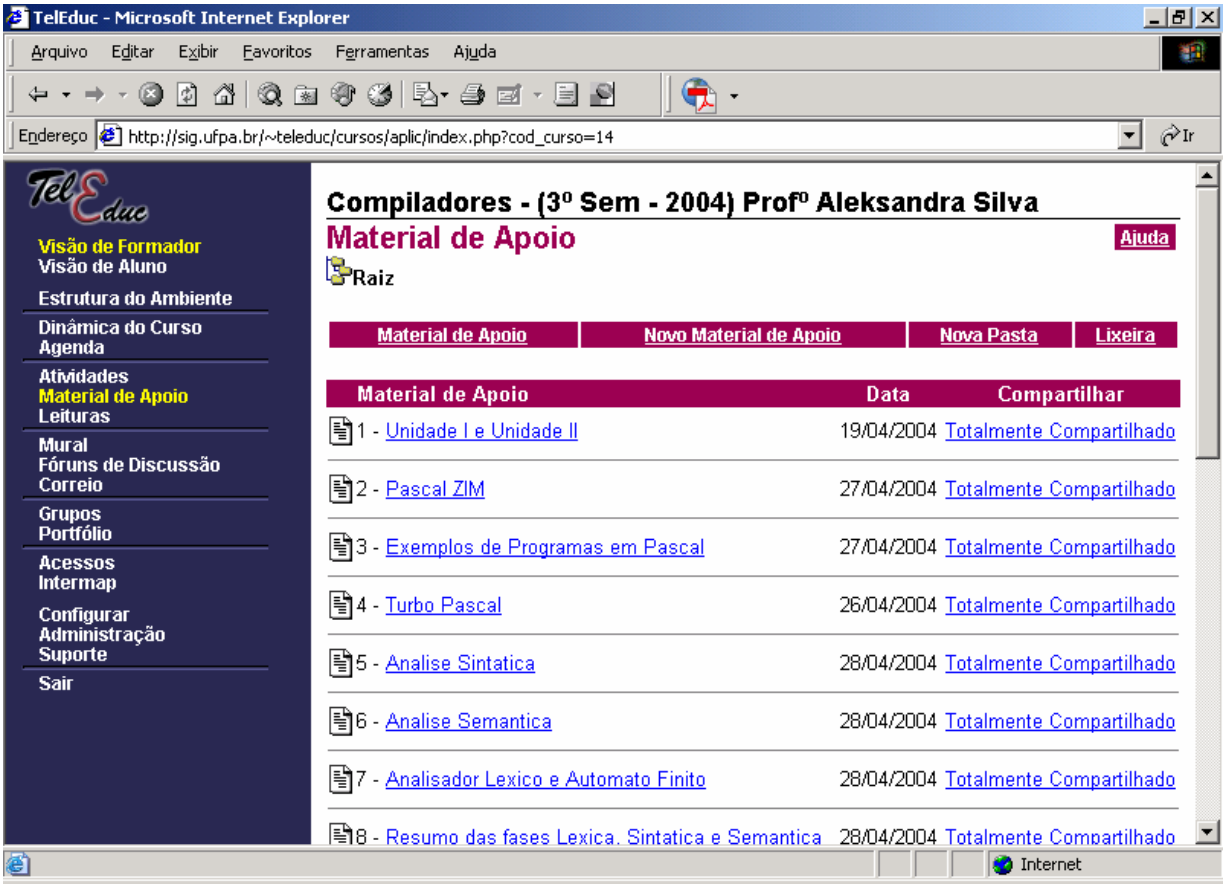


Figura 1. Recurso denominado "Material de Apoio", no TelEduc.

No planejamento dos projetos de compiladores, sugerimos, do ponto de vista do professor, que os seguintes aspectos sejam observados.

- o projeto precisa ser factível no tempo da disciplina, por estudantes sem experiência prévia na construção de compiladores;
- a linguagem a ser compilada deve ser uma linguagem restrita. É aconselhável que o professor não especifique recursos ou interfaces para o compilador que comprometam, com detalhes de implementação, o desenvolvimento incremental e o tempo de execução do projeto;
- o projeto especificado deve ser altamente modular, de modo que a solução dos estudantes possa ser substituída por outra (fornecida pelo professor) e deve ser tal que os estudantes possam utilizar uma infra-estrutura de código fornecida pelo professor. Em nossa experiência, a dependência entre os módulos é minimizada implementando-se a comunicação entre os módulos a partir de arquivos em memória secundária;
- o projeto do compilador deve ser instrumentalizado com porções de código, freqüentemente incompletas, mostrando representações intermediárias de programas, de modo que auxilie os estudantes a compreender o funcionamento interno do compilador;
- módulos reutilizáveis são descritos e apresentados como exemplos de sala de aula, sem fornecer soluções completas, mas de modo que possam ser utilizados em outros projetos nos períodos subseqüentes;

No contexto do estudante, é importante que o processo de desenvolvimento do projeto do compilador seja especificado e descrito pelos estudantes em documento padrão de Especificação de Requisitos de Software (ERS). A arquitetura e o código devem exemplificar conceitos estudados na engenharia de software e a documentação do projeto deve ser rastreável;

Com a utilização dos recursos disponíveis em ambientes interativos de aprendizagem, foi possível observar que os projetos de compiladores podem ser amadurecidos, compartilhados e aperfeiçoados por diferentes instituições. Além disso, os estudantes se sentem encorajados a publicar gratuitamente seus projetos, o que favorece o sentimento, nos estudantes, de que a disponibilização de seus projetos gratuitamente na rede pode gerar novas linguagens para a comunidade. Adicionalmente, com o auxílio das mensagens postadas nas ferramentas de comunicação, a documentação das ferramentas utilizadas é aperfeiçoada utilizando-se um recurso de "Perguntas mais Freqüentes".

Outra consideração diz respeito às horas didáticas do professor. No sistema educacional tradicional, essas horas freqüentemente são insuficientes para os estudantes. Entretanto, para os estudantes que possuem acesso à Internet, ferramentas de colaboração e pesquisa, as facilidades disponíveis podem ser oportunas e convenientes para estudar o material no seu próprio ritmo.

Os ambientes virtuais de aprendizagem permitem a incorporação de *Applets* Java[4] que podem ajudar a criar um ambiente interativo de "aprender fazendo", através de simuladores (no Laboratório de Compiladores[18], *applets Java* são utilizados para simular o comportamento de analisadores léxicos e sintáticos como autômatos). Além de demonstrar conceitos, esses recursos podem aumentar a motivação e instigar um maior interesse entre estudantes.

O compartilhamento é substancialmente potencializado com a utilização dos recursos em rede. Um típico projeto de compilador utiliza muitas estruturas de dados "padrões" e possui algum nível de código repetitivo e de baixo nível (ex. *templates* para as instruções *assembly*). Segundo as diretrizes curriculares, a matéria Compiladores deve ser precedida do estudo de conceitos teóricos de linguagens e autômatos, sistemas operacionais e arquiteturas de computadores. Entretanto, presumivelmente os estudantes tiveram um curso de estruturas de dados (ou pesquisa operacional) antes do curso de compiladores e podem, já possuir, componentes que podem ser utilizados no projeto do compilador (estrutura *hashing*, por exemplo). Com os recursos de compartilhamento existentes em ambientes interativos de aprendizagem, esse código de apoio pode estar disponível e documentado em uma área compartilhada e pode ser reusado em diferentes projetos. Fornecer uma área compartilhada para o código de apoio (incluindo toda a descrição desse código e de suas interfaces) fornece um nível moderado de abstração, removendo uma grande porcentagem de possíveis armadilhas, além de permitir que os estudantes focalizem seus esforços nos aspectos mais importantes do projeto.

Uma área de compartilhamento é fundamental, para que os estudantes realizem as reflexões necessárias sobre as possíveis soluções no projeto de compilador. Se os estudantes tiverem uma única opção de recurso, dificilmente realizarão essas reflexões. O compartilhamento dos projetos permite a avaliação e a comparação entre os projetos desenvolvidos. Da mesma forma, a disponibilização de porções intermediárias de código enriquecem as opções para os estudantes. Segundo Rivas [22], a riqueza de opções é de grande importância, constituindo-se uma medida do nível de estruturação do curso. Indubitavelmente, a riqueza de alternativas para conceitos e ferramentas constitui-se em um valioso arsenal para o estudante.

A percepção dos aprendizes é valorizada com a utilização dos recursos telemáticos. Permite aos estudantes compreenderem as atividades de outros e ajustarem suas próprias atividades através da reflexão sobre os resultados alcançados.

Na experiência com o TelEduc, a percepção dos aprendizes foi valorizada pela abordagem chamada *feedback* compartilhado [27], que consiste em coletar e apresentar para a comunidade, de forma automatizada, informações sobre as atividades individuais dos aprendizes, dentro de um espaço de trabalho compartilhado. Esse espaço, no TelEduc, foi construído com a utilização dos portfólios individuais e de grupos, que permitiram transmitir um senso de atualização contínua das ações individuais e o progresso global da turma. A grande vantagem de utilizar essa ferramenta está no fato de que os estudantes estão comunicando suas atividades a outros, refletindo, dessa forma, sobre suas ações e permitindo que os demais possam fazer comentários sobre elas e observar as conseqüências das ações efetuadas. Os portfólios representam espaços (pastas e arquivos) onde os estudantes postam os artefatos (códigos, documentação, etc.) produzidos nas fases do projeto. Esses artefatos podem ser comentados por professores ou por outros estudantes, desde que o autor tenha compartilhado o artefato.

6. CONCLUSÕES

Os AVAs têm sido propostos para apoiar comunidades virtuais de aprendizagem, visando a construção coletiva de conhecimento. Este artigo relatou a experiência dos autores no ensino de compiladores e reflete as idéias desenvolvidas através dessa experiência, favorecendo a utilização de ambientes virtuais de aprendizagem como forma de apoiar o processo de construção do conhecimento e o desenvolvimento dos projetos. É improvável, entretanto, que a experiência de aprender sobre compiladores com apoio telemático seja única. Apesar disso, por se tratar de uma disciplina onde a abordagem de projetos freqüentemente se mostra mais adequada, acreditamos que a

[4] no Laboratório Virtual de compiladores, utilizamos *applets* Java para simular o comportamento do analisador léxico, representando as transições de estados no autômato.

experiência com a disciplina de compiladores permita a especificação de requisitos e o projeto de ambientes virtuais de aprendizagem que valorizem a percepção, o reuso de conhecimento e facilitem o planejamento de cursos por meio do compartilhamento de recursos, até mesmo entre grupos de professores e aprendizes de diferentes instituições.

A experiência com o TelEduc também revelou diversos problemas ainda não tratados pelos ambientes virtuais de aprendizagem. Talvez, a priori, as principais dificuldades aparentem estar integração do conhecimento: no TelEduc, por exemplo, quando um aluno decide abandonar o espaço da disciplina de Compiladores e acessar, por exemplo, a disciplina de Linguagens Formais, é necessário fornecer novamente sua identificação no sistema. Assim, reproduz-se a idéia de "paredes" virtuais, que impedem a transição natural do estudante entre os objetos de aprendizagem, que são tão intrinsecamente relacionados, como é o caso dessas duas disciplinas. Da mesma forma, do ponto de vista dos professores, a visão que os atuais AVAs fornecem dos estudantes reduz-se sumariamente aos instantâneos tirados a partir de seu desempenho exclusivamente nas disciplinas ministradas por aquele professor. Um professor de compiladores, por exemplo, não tem a sua disposição, os relatórios do desempenho de estudantes em outras disciplinas, o que poderia facilitar sua visão de avaliação como instrumento de acompanhamento. Além disso, a utilização dos AVAs, como atualmente projetados e especificados, aliados a abordagens pedagógicas tradicionais, sobrecarregam as atividades de mediação do processo de aprendizagem desanimando a utilização desses recursos.
Os problemas levantados neste artigo contribuem para especificação de requisitos [23] coletados após a experiência dos autores em diferentes instituições e com diferentes ambientes. A continuação deste trabalho consiste, portanto, em desenvolver um ambiente com componentes de software que atendam aos requisitos especificados. Neste ambiente, buscamos viabilizar a integração dos conteúdos e das atividades executadas por um aprendiz, independentemente da comunidade (turma ou curso) que participa, evitando, através de uma base de artefatos compartilhados, o isolamento de disciplinas e o isolamento entre as diferentes etapas da aprendizagem. Adicionalmente, é possível incorporar objetos de aprendizagem, como *applets* (o que parece ser bastante interessante em um curso de compiladores), sem que seja necessário realizar modificações nas especificações de código do AVA.

Referências

[1] Acebal, C. F., Castanedo, R.I.., Lovelle, J.M.C. *Good Design Principles in a Compiler University Course.* ACM SIGPLAN Notices, April 2002. p. 62 – 73.

[2] Aho, A.V., Sethi, R., Ullman, J.D. *Compilers: principles, techniques, and tools*. Massachusetts: Addison Wesley Publishing Co., 1986.

[3] Aiken, A. Cool: A Portable Project for Teaching Compiler Construction. *ACM Sigplan Notices* 31(7), pages 19-26, July, 1996. Disponível em <http://theory.stanford.edu/~aiken/publications/papers/sigplan96.ps>

[4] Appel, A., Ginshurg, M. *Modern compiler Implementation in C.* Austrália: Cambridge University Press, 1988.

[5] Baldwin, D. A Compiler for Teaching about Compilers. Proceedings of the *Technical Symposium on Computer Science Education* 34th SIGCSE'03 Technical Symposium On Computer Science Education, fev. 2003. p. 220-223. Reno, Nevada, USA. ISSN:0097-8418. Disponível em <http://doi.acm.org/10.1145/611892.611974>.

[6] Ben-Ari, M. Constructivism in computer science education. In: *Journal of Computers in Mathematics and Science Teaching,* 20(1), 2001, 45–73.

[7] Ben-Ari, M. Situated learning in computer science education. *Computer Science Education* (aceito para publicação). Disponível em: <http://stwi.weizmann.ac.il/g-cs/benari/articles/sit-cs.pdf>. Acesso em 21 mai 2004.

[8] Ben-Ari, M. *Constructivism in Computer Science Education*. Proceedings of the twenty ninth SIGCSE Technical Symposium on Computer Science Education, 1998, P. 257 - 261.

[9] Brito, S.R., Tavares, O.L., Menezes, C.S. Um ambiente virtual para aprendizagem de compiladores com suporte inteligente à mediação. In: Internacional. In: *Conference on Engineering and Computer Education*, 2003, Santos. ICECE'2003. COPEC, 2003.

[10] Christopher, W., Procter, S., Anderson, T. The Nachos instructional operating system. In: *Winter USENIX Conference*. pages 479-488. January, 1993.

[11] Favero, E.L. *Programação em Prolog: Uma abordagem prática.* Editora da UFPA. (em publicação, 2004)

[12] Furtado, O. J. V. "O ensino de Linguagens Formais vinculado ao ensino de Compiladores". In: XI Workshop de Educação em Computação. Disponível em: <http://bioinfo.cpgei.cefetpr.br/anais/SBC2003/pdf/arq0056.pdf>. Acesso em 15 dez 2003.

[13] Grégoire, R.; Laferrière, T. "Project-based collaborative learning with network computers: teacher's guide". *Canada's Schoolnet*, 2001. Disponível em: <http://www.tact.fse.ulaval.ca/ang/html/projectg.html#anchor387673>. Acesso em: 05 mai. 2001.

[14] Hadjerrouit, S. A constructivist approach to object-oriented design and programming. Proceedings of the *4th annual SIGCSE/SIGCUE on Innovation and technology in computer science education* , 1999, p. 171 - 174.

[15] Harb, M. P. A. A., Brito, S. R., Silva, A. S., Favero, E. L., Tavares, O. L., Francês, C. R. L. *AmAm: ambiente de aprendizagem multiparadigmático*. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 2003, Rio de Janeiro. SBIE'2003. Rio de Janeiro: NCE-IM-UFRJ, 2003. v. 14, p. 223-232.

[16] Holland, D.; Lim, A.; Seltzer, M. *A New Instructional Operating System*. Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education, Mar. 2002. p. 111 – 115.

[17] Kaplan, A; D. Shoup. *CUPV — A Visualization Tool for Generated Parsers.* Proceedings of the 31st SIGCSE Technical Symposium on Computer Science Education, Mar. 2000. p. 11 –15.

[18] LABCOMPL. *Laboratório Virtual de Compiladores*. Disponível em: <http://www.inf.ufes.br/~tavares/labcomp2000/>. Acesso em 10 jan 2004.

[19] Lévy, P. *As Tecnologias da Inteligência*. Rio de Janeiro: Editora 34, 1993.

[20] Machado, F. B., Maia, L. P. *Arquitetura de Sistemas Operacionais*. 2 ed . Rio de Janeiro: LTC, 1997.

[21] MEC. *Diretrizes Curriculares de Cursos da área de Computação e Informática*. Disponível em: <http://www.mec.gov.br/sesu/ftp/curdiretriz/computacao/co_diretriz.rtf>. Acesso em 11 dez 2003.

[22] Rivas, L. G. R. *Some Thoughts on the Teaching of Informatics Engineering*. Disponível em <http://luisguillermo.com/english/default.htm>. Acesso em 10 jan 2004.

[23] Silva, A.S.; Brito, S.R.; Favero, E.L.; Hernández-Domínguez, A. Tavares, O.L.; Francês, C.R.L. *Uma arquitetura para desenvolvimento de ambientes interativos de aprendizagem baseado em agentes, componentes e framework*. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO. SBIE'2003. Rio de Janeiro: NCE-IM-UFRJ, 2003. v. 14, p. 203-212.

[24] Slonneger, K; Kurtz, B. L. *Formal Syntax and Semantics of Programming Languages: A Laboratory-Based Approach.* Addison-Wesley, NY, 1995.

[25] Tavares, O. L, Brito, S. R., Souza, R. S. ; Menezes, C. S. Ambiente de apoio à mediação de aprendizagem: Uma abordagem orientada por processos e projetos. *Revista de Informática na Educação*, set., 2001, p. 77-87.

[26] TelEduc. *Um ambiente de Ensino a Distância*. Disponível em: http://teleduc.nied.unicamp.br/teleduc/. Acesso em 15 jan 2003.

[27] Togneri, D. F., Falbo, R. A., Menezes, C. S. Supporting Cooperative Requirements Engineering with an Automated Tool. In: Workshop on Requirements Engineering, 5., 2002, Valência – Espanha. Anais… Valência: CYTED, 2002.

El Desarrollo Académico de la Computación en la Argentina y la cooperación Latinoamericana

Jorge Aguirre[1]
Universidad Nacional de Río Cuarto
jaguirre@dc.exa.unrc.edu.ar

Abstract

The Argentine scientific system reached such an important development in the XXth century, that three argentine scientists obtained Nobel prizes, two of which because of the work developed in local research institutes. That was not so, in the field of Computer Sciences. Ten years ago, this was actually an area of vacancy, so much as there were only a pair of doctoral graduates living in the country, while the university population was of about five thousands students. On the other hand, research on Computer Sciences had started very early, around 1957. Only ten years after the first computers were born, the earliest research groups arose and very soon their results began to appear in qualified international media. There were also important initiatives during the following forty years, but most of them were unsuccessful and until the last decade there were not globally significant changes. An explanation for this can be found in the frequent interruptions of democracy by military coups, the lack of state policies and the tendency of argentine leadership to abandon the work initiated by its predecessors. These aspects had a remarkable effect in a discipline that was in the beginnings. This work describes this story of advances and withdrawals, the improvement projects that were set up and the results obtained. The focus is on the course of Computer Sciences in the Academia in Argentina, but historical and social context is also considered, because if not, it would be unable of being understood. Links with other Latin American countries are also mentioned.

Keywords: Computer Sciences History, Latino American cooperation, High level education in Computer Sciences, Computer Sciences researchs.

Resumen

El sistema científico argentino ha conseguido alcanzar un importante desarrollo durante el siglo XX, desarrollo que permitió que tres argentinos obtuvieran el Premio Nobel, dos de ellos por trabajos desarrollados en institutos locales. No ha sucedido lo mismo en el Campo de las Ciencias de Computación. Éstas, diez años atrás constituían una verdadera área de vacancia, al punto que sólo había un par de doctores en Computación en el país, contrastando este hecho con el tamaño de la matrícula universitaria, que llegaba a casi 5000 alumnos. Sin embargo la investigación en computación se inició bastante temprano en la Argentina. En 1957, a sólo una década del nacimiento de la Computadora, ya se habían constituido los primeros grupos de investigación, que poco después comenzaron a publicar en medios internacionales calificados. También hubo emprendimientos importantes a lo largo de los 40 años que siguieron, pero casi todos ellos quedaron truncos. Así, hasta hace pocos años no se habían producido cambios globales significativos en el estado de la disciplina. Esto se explica por la secuencia de gobiernos militares que interrumpieron la democracia argentina, la falta de políticas de estado y la tendencia de los gobernantes locales a abandonar las obras iniciadas por sus predecesores. Motivos que tuvieron especial efecto sobre una disciplina aún incipiente. Aquí se analiza esta historia de avances y retrocesos, los proyectos de mejora emprendidos y sus resultados. El análisis se centra en el devenir de la Informática académica en la Argentina, pero se hace referencia a su contexto histórico y socio económico, sin lo cual sería imposible entenderlo. También se tratan sus vinculaciones con otros países latinoamericanos: La colaboración con Uruguay, los proyectos conjuntos con Brasil, y la repercusión en Latinoamérica y el Caribe de la Escuela Superior Latino Americana de Informática.

Palabras clave: Historia de la Computación, Cooperación Latinoaricana, Educación Superior de Informática, Investigación en Ciencias de Computación

[1] Este trabajo ha sido desarrollado en el marco de proyectos subsidiados por la Agencia Córdoba Ciencia y la SCyT de UNRC.

1. Introducción.

Existen varios trabajos sobre el marco histórico académico en que se inició la Computación en la Argentina, como el de Manuel Cereijido, que describe la época e influencia del primer argentino que obtuvo el premio Nobel [6], el de A. Dávalos, sobre el Dr. Balseiro y la investigación y enseñanza de la Física Nuclear [1] y otros ([5, 8, 9]). También se cuenta con algunos trabajos sobre la historia de la Computación en la Argentina ([3, 12]) y de su vinculación con otros países latinoamericanos ([7, 11, 13]). No obstante, algunos de ellos no cubren el período más reciente, otros desconocen algunos hechos o son demasiado sucintos. Esto, sumado a mi participación personal en muchas de los acontecimientos de esta historia y a mi convencimiento de que la experiencia colectiva es fundamental para interpretar y conducir todo proceso social, en particular los de enseñanza superior que nos competen como profesores universitarios, me hizo sentir la obligación de escribir este artículo y presentarlo en el CIESC.

La Argentina cuenta con un extenso sistema científico y tecnológico, constituido por universidades e institutos; con un ente de promoción científica nacional, el CONICET que soporta una exigente y nutrida Carrera de Investigador, cuyos miembros se desempeñan tanto en universidades como en institutos. También hay algunos entes provinciales de promoción científica. Dentro del sistema hay numerosos centros de excelencia. Tres egresados de las universidades argentinas han sido galardonados con el Premio Nobel: Bernardo Houssay en 1947 ([6]), Luis Federico Leloir en 1970 ([6]) -ambos por trabajos realizados en el país- y César Milstein en 1984, por trabajos realizados en Gran Bretaña. ([8]).

Sin embargo las Ciencias de Computación han permanecido muy rezagadas; hace diez años constituían una verdadera área de vacancia, al punto que sólo había un par de doctores en Computación en el país. Situación que contrastaba con la gran cantidad de alumnos cursando carreras universitarias de Informática (aproximadamente 5.000). Esto sucedía a pesar de que la Computación tuvo en el país un temprano y prometedor inicio, producido a sólo diez años del nacimiento de la Computadora. Así, en 1957 se formaban los primeros institutos y la primera computadora, "Clementina", ingresaba a la Universidad de Buenos Aires (UBA) en 1960. Pero este temprano desarrollo se interrumpiría en 1966, pocos días después de la toma del gobierno por la cúpula militar, cuando el presidente de facto, Gral. Onganía intervino las universidades nacionales y la policía entró a la Facultad de Ciencias Exactas de la Universidad de Buenos Aires -durante la luego llamada "Noche de los bastones largos"-. A partir de entonces ha habido períodos de letargo, emprendimientos importantes de desarrollo académico y tecnológico, algunos regionales y otros binacionales, avances y retrocesos, hasta llegar a una situación actual prometedora pero frágil, que espero que la experiencia colectiva ganada nos ayude a impedir que se transforme en una nueva frustración.

Este artículo analiza esa historia de avances y retrocesos. El foco es el devenir de la Informática en el sistema científico y tecnológico argentino, pero se hace referencia a su contexto histórico y socio económico, sin cuyo conocimiento sería vano tratar de entenderlo; también se hace referencia a sus vinculaciones con otros países latinoamericanos. La organización de este trabajo sigue las etapas históricas más importantes hasta 1999, cada una de las cuales cuenta con una sección. Dentro de ellas se describen los proyectos fundamentales, su origen, fin y consecuencias. Así, son tratados: la fundación de los primeros institutos dedicados a la Computación y la introducción de la primera computadora; el nacimiento y la evolución de la enseñanza de grado y de posgrado; los proyectos industriales de construcción de computadoras de los 70; el Programa Nacional de Informática y Electrónica del gobierno del 84 y sus proyectos PABI, EBAI – ambos en cooperación con Brasil- y la ESLAI –para Latinoamérica y el Caribe-; por último el proyecto FOMEC de 1994. Finalmente se analiza la situación actual y se extraen conclusiones.

La información se ha obtenido de la bibliografía citada, de consultas a los protagonistas y de registros del autor. Se ha tratado de dar a la presentación de los distintos temas la mayor objetividad, aunque este análisis no puede escapar a la subjetividad que impregna a la interpretación histórica reciente. Tampoco puede rehuir la calificación de crónica, ya que su autor escribió sus primeros programas en Autocode para Clementina, sufrió los bastonazos del 66, siendo aún estudiante, y luego fue protagonista de muchos de los sucesos narrados: el grupo del Observatorio Nacional de Física Cósmica, la fundación de las carreras de computación en San Luis y en Río Cuarto, los proyectos de producción de computadoras nacionales de los 70, el PABI, la ESLAI, el FOMEC y el doctorado en Ciencias de Computación de la Universidad de Buenos Aires.

2. Los tiempos Previos.

2.1 La Reforma Universitaria.

Al comenzar el siglo XX el sistema universitario argentino estaba conformado por la Universidad de Córdoba, una de las más viejas de la colonia y la Universidad de Buenos Aires. Por esa época, se incorporó la Universidad de La Plata, que tuvo un importante desarrollo en Ciencias Exactas a partir de la incorporación de varios destacados científicos alemanes ([9]). Las universidades habían pasado de depender de la Iglesia a depender del gobierno que

decidía las designaciones de profesores y las cuestiones académicas fundamentales. Su organización respondía a la ideología imperante que las veía destinadas a formar una reducida elite dominante, de un país agroexportador.
En 1918, durante el gobierno de Hipólito Yrigoyen, en la ciudad de Córdoba nació un importante movimiento estudiantil, que veía a la Universidad como promotora de progreso y cambio social y propiciaba importantes cambios en la estructura universitaria. Este movimiento adquirió el nombre de "Reforma Universitaria". Los cambios deseados se basaban en los siguientes principios básicos: a) la autonomía de las universidades del poder político; b) El gobierno de las mismas por consejos integrados por representantes de tres claustros: el de docentes, el de graduados y el de estudiantes; c) La asignación de cargos docentes mediante concursos públicos; d) La libertad de cátedra, que daba total libertad de pensamiento y difusión a los que ganaran la titularidad de una cátedra, permitiendo la coexistencia de cátedras paralelas con distintas orientaciones ideológicas o académicas; e) La asignación por parte del estado de presupuestos que permitieran el adecuado funcionamiento universitario. Luego de una larga huelga y gran movilización que se extendió a otras universidades. los estudiantes lograron que el gobierno promulgara la "ley de Reforma Universitaria".
La Reforma Universitaria –que en cierta forma anticipó el mayo francés del 68- dió a las universidades argentinas un estructura muy moderna que permitió su desarrollo e integró a todos los sectores universitarios en la responsabilidad de su conducción.

2.2 Del gobierno del Gral. Perón a la situación en que ingresa la Computación a la Universidad.
Al iniciarse la posguerra, en 1945, fue elegido Presidente el General Juan Domingo Perón, líder del Movimiento Justicialista. El gobierno de Perón introdujo importantes mejoras laborales y sociales. También impulsó el desarrollo industrial, para lo cual cerró la economía dejando en el estado la decisión de qué productos se podía importar. Mantuvo una política autónoma , no falta de conflictos, frente a Estados Unidos, que pugnaba por asumir su nuevo rol de liderazgo occidental. También se enfrentó a la clase terrateniente que había tenido predominio en la conducción del país.
No obstante, mantuvo un importante control sobre la prensa y las posibilidades de expresión y quedó opuesto al movimiento reformista y a la izquierda. Respecto de la ciencia y la tecnología el gobierno de Perón, por un lado auspició la introducción de tecnología de punta, impulsó la industria, que logró producir automóviles, máquinas agrarias, equipos electrodomésticos y hasta llevó al país a ser uno de los primeros en producir aviones a reacción (los modelos de cazas Pulqui I y II). También, durante su gobierno se fundó la Comisión Nacional de Energía Atómica y el Instituto de Física Atómica de Bariloche [1, 5] (más tarde llamado Instituto Balseiro).
Mientras que por otro lado se dejó cesantes a muchos profesores universitarios resintiéndose el avance de las universidades ([6]).
Muchos y heterogéneos sectores se unieron en contra de Perón y en 1955, durante su segundo período constitucional, luego del fracaso de un levantamiento de la Aeronáutica Naval, que el 16 de junio bañó en sangre la histórica Plaza de Mayo, bombardeando y ametrallando a una concentración civil que manifestaba su adhesión al gobierno, fue derrocado por un golpe militar auto denominado "Revolución Libertadora". Luego de una breve transición asumió la presidencia el Gral. Pedro Eugenio Aramburu. El movimiento justicialista no perdió vigencia, mientras que sus adherentes sufrieron represiones que llegaron hasta el fusilamiento de obreros. Perón desde el exilio continuó siendo el líder del movimiento y su peso político fue recuperando protagonismo con el paso de los años.
El gobierno de Aramburu intervino las universidades y hacia 1957, estas se normalizaron eligiendo sus autoridades según la ley de Yrigoyen. Las autoridades universitarias electas incluían a brillantes universitarios que iban a impulsar una progresista e importante transformación. Esta será la Universidad en que ingresa la Computación. Universidad, que durante las guerras que devastaron Europa del 36 al 45, se había enriquecido con la incorporación de importantes científicos europeos. La radicación de Rey Pastor en la Argentina, primero como profesor de la Universidad de Rosario y luego de la UBA, había dado origen a la Escuela Matemática Argentina a la que se integraron inmigrantes ilustres como Beppo Levi –en la Universidad de Rosario- y jóvenes brillantes como Luis Santaló y Manuel Balanzat en Buenos Aires. Por otra parte la Física también había tenido un significativo desarrollo, particularmente en investigaciones nucleares. Este estado académico permitió a las nuevas autoridades de las Universidades Nacionales impulsar importantes proyectos de desarrollo de las Ciencias Exactas. Resulta emblemático el de la Facultad de Ciencias Exactas y Naturales de la UBA cuyo Decano era el Dr. Rolando García y cuyo Vicedecano era el Dr. Manuel Sadosky . La Facultad creció rápidamente: se ejecutó un importante proyecto de formación de recursos humanos, se consiguió equipamiento e infraestructura y se incrementó notablemente la planta de docentes-investigadores que se calificaba con el retorno los que partían a realizar estudios de posgrado en el exterior. En este contexto de ebullición académica iba a ingresar la primera computadora en la Academia rioplatense.

En el ámbito nacional al gobierno de Aramburu siguió el del Dr. Arturo Frondizi, que ganó las elecciones de 1958. Estas elecciones fueron llamadas por el gobierno militar con la proscripción del movimiento justicialista, cuyos votos decidieron el triunfo de Frondizi. El nuevo gobierno civil apoyó el desarrollo de las universidades nacionales, que continuaron el camino iniciado en el 57.

2.3 El procesamiento mecánico de datos previo a la Computación.
Antes del nacimiento de la computadora ya existía en la Argentina un importante desarrollo de lo que se denominaba "procesamiento mecanizado de datos". Este se realizaba con máquinas de registro directo: tabuladoras, clasificadoras y perforadoras. Los datos se registraban en tarjetas perforadas, que constituían el único medio de memoria. El proceso se realizaba mediante una sucesión de etapas, constituidas por el paso de lotes de tarjetas; sucesión iniciada por los datos de partida y seguida generalmente por otros lotes conteniendo información intermedia, obtenida en distintas etapas del proceso. Cada etapa daba como resultado el reordenamiento del lote procesado, uno nuevo, o una impresión. Las máquinas eran capaces de realizar operaciones aritméticas sobre los datos representados en una tarjeta y de tomar decisiones lógicas simples a partir de ellos. Las operaciones que cada máquina debía realizar se programaban mediante la conexión (por cableado) de contactos de un tablero. Algunos organismos del estado y las grandes empresas nacionales argentinas y uruguayas contaban con equipamiento de este tipo, algunos de los cuales perduraron hasta mucho después del auge de la computación –hasta ya entrados los 80-.

3. Entrada de la Computación en la Universidad.

El Dr. Sadosky en 1957 inició los trabajos de implantación de la Computación en la Facultad de Ciencias Exactas y Naturales de la UBA, e impulsó la adquisición una computadora. Se seleccionó a una Mercury Ferranti que llegó al país en 1960 y fue instalada en el flamante Pabellón I de la Ciudad Universitaria en construcción[2]. La Computadora recibió el nombre de Clementina Sus dimensiones sorprenden hoy: ocupaba toda una sala, estrictamente acondicionada; pero su memoria principal tenía sólo 1 K palabras de 48 bits. Como memoria secundaria tenía tambores magnéticos y la entrada/salida se realizaba mediante cinta de papel perforada, impresora, consola y un parlante con el que deslumbraba tocando algunas oberturas. Contaba con un compilador de un lenguaje orientado al cálculo, Autocode, con el que se iniciaron las primeras camadas de programadores argentinos. En 1962 Sadosky fundó el Instituto de Cálculo, que dotado de la nueva herramienta, se ubicó en la primera línea del acelerado desarrollo de la Facultad, mandando a varios de sus jóvenes integrantes a realizar estudios al exterior y alcanzando masa crítica y reconocimiento rápidamente. Clementina permitió iniciar investigaciones de desarrollo de software de base, de desarrollo de periféricos e interfaces, de matemática aplicada y permitió realizar transferencias en distintas áreas.
En la Facultad de Ingeniería de la UBA también se constituyeron grupos de investigación y desarrollo en Computación y el Ing. Humberto Ciancaglini fundó un grupo de electrónica digital que llegó a diseñar y construir un prototipo de computadora, que fue llamada CEFIBA (1962). En la Universidad Nacional del Sur también se iniciaron trabajos en Computación digital y el grupo del Ing. Jorge Santos llegó a construir una computadora denominada CENUS (1962) a la que sólo le faltó (por falta de presupuesto para construirla) la memoria suficiente para que pudiera funcionar.
El proceso vertiginoso de avance de la estructura científica de los años de 1958 a 1966 fue acompañado por una gran politización estudiantil, mientras que el poder político de la cúpula militar crecía y cuestionaba constantemente las acciones del gobierno nacional, a través de lo que en la época se conoció como "planteos militares". Finalmente, en 1962, el Dr. Frondizi fue conminado a renunciar y ante su negativa apresado y mantenido prisionero en una isla. Se implementó una parodia de gobierno civil, bajo control de los militares y finalmente, luego de una cruenta confrontación de dos bandos militares opuestos, se llamó en 1963 a elecciones nacionales, de cuyo proceso nuevamente se excluyó al movimiento justicialista.

3.1 La noche de los bastones largos.
En las elecciones de 1963 fue elegido presidente el Dr. Arturo Illia. Durante la presidencia de Illia, continuaron los planteos militares y finalmente su gobierno fue depuesto por los comandantes de las tres fuerzas armadas. Asumió la presidencia el general Juan Carlos Onganía. Los estudiantes fueron el sector que más se opuso, si no el único, a la nueva dictadura militar. En este contexto, sin el apoyo de las fuerzas populares y en la mira de la dictadura, los días de la Universidad estaban contados. El 26 de julio, Onganía firmó el decreto de intervención a las Universidades Nacionales. Esa "Noche de los bastones largos", antes de ser notificadas las autoridades universitarias, fuerzas de asalto policiales irrumpieron a bastonazos en la vieja sede de la Facultad de Ciencias Exactas de la UBA –hoy Manzana de las Luces- en la que deliberaban autoridades, profesores y estudiantes. Nadie escapó a los bastonazos y muchos fueron detenidos, incluso un profesor visitante estadounidense. Indignados, los profesores de la Facultad renunciaron masivamente de inmediato, iniciándose el éxodo de nuestros más destacados investigadores en Ciencias Exactas. El proyecto había sido truncado, un sablazo había bastado para cortar el hilo de su historia. Después de varios meses de clausura, la Universidad reabrió sus puertas, mientras renacían las protestas estudiantiles, su represión y las detenciones. Era otra Universidad. La policía ocupaba aulas y pasillos. Estaban prohibidas las reuniones y en la Facultad de Ciencias Exactas de la UBA casi no quedaban profesores formados. Algún tiempo

[2] El proyecto de la Ciudad Universitaria de la UBA quedaría luego trunco, llegándose a construir solamente tres pabellones, el I destinado fundamentalmente a investigación que alberga los departamentos de Física, Matemática y Computación de la Facultad de Ciencias Exactas, el II destinado a docencia y a los demás departamentos de la misma Facultad y el III destinado ala Facultad de Arquitectura

después, Clementina también dejaría de funcionar, exhausta ante la falta de mantenimiento. El centro en que naciera la computación argentina pasaría muchos años hasta volver a tener una computadora[3].

4. La dictadura militar de 1966 a 1973, auto denominada "Revolución Argentina".

El gobierno militar presidido por Onganía y luego por los generales Roberto Marcelo Levingston y Alejandro Agustín Lanusse se extendería hasta el 73. Durante este período las conducciones de las universidades nacionales se limitaron a mantener su funcionamiento. Sólo algunos profesores aislados introdujeron algunas innovaciones, como los Ingenieros Esteban Ditada y Luis Trab que comenzaron a dictar Teoría de Lenguajes Formales y Autómatas y Diseño de Compiladores en la Facultad de Exactas de la UBA. Los militares evitaron dar trascendencia a las actividades universitarias, considerándolas potencialmente peligrosas, ya que el estudiantado podía crearles resistencia. Los militares imaginaban posibles escenarios de conflicto en Latinoamérica por lo que no descartaban la necesidad de realizar investigación y desarrollo orientados a la seguridad nacional, pero pensaban que ella debía realizarse en institutos separados de las universidades. Las universidades quedaron relegadas a su rol docente, empobrecido por la falta de investigación e iniciativa.

4.1 Un intento de reparación de Onganía.

Durante el gobierno de Onganía renació un instituto, ubicado en la localidad de San Miguel (a 30 Km de Buenos), que pertenecía a la Compañía de Jesús y había tenido importancia en la época del proyecto de nuclear de Richter de los años 50 [5]. En él se había construido, en aquella época, un acelerador de partículas, una cámara de niebla y se había establecido la primera fábrica nacional de Contadores Geiger. Luego había quedado prácticamente vacío, ocupado por un par de astrónomos jesuitas. En 1969, Onganía, influenciado por el Dr. Mariano Castex, en ese tiempo sacerdote y su asesor espiritual, intentó revertir la imagen de destructor de la ciencia que se había granjeado en la Noche de los Bastones Largos, creando la Comisión Nacional de Estudios GeoHeliofísicos (CNEGH) cuyo centro más importante pasó a ser el mencionado instituto, convertido en el Observatorio Nacional de Física Cósmica de San Miguel (ONFCSM). Al ONFCSM se integraron una importante cantidad de científicos repatriados, la mayoría investigadores que habían renunciado en el 66. Se conformaron importantes grupos en una variedad de disciplinas (variedad que trascendía ampliamente la denominación del Centro). Así se constituyeron grupos de: Física del Plasma, liderado por Carlos Abeledo, Daniel Winivesky y Enrique Distefano; Semi-Conductores liderado por Ivan Chambuleirón; Física Solar, Geofísica, Electrónica Aplicada y Matemática Aplicada, este último liderado por Pedro Elías Zadunaisky. Este último departamento incluyó un grupo de Ciencias de Computación que contó con el aporte de jóvenes que luego iban a adquirir gran relevancia, como Armando Haeberer y Eduardo Sontag – colaboradores contratados- y Gregorio Chaitin – visitante-. En el grupo se iniciaron actividades de investigación y desarrollo en lenguajes y compiladores produciéndose un primer compilador en 1972. En 1975 el grupo fue seleccionado para diseñar e implementar el compilador del proyecto más importante de fabricación de computadoras nacionales, del que se hablará en la próxima sección .

4.2 El deebilitamiento de la dictadura.

El descontento popular fue creciendo con los años. La represión obligó a la oposición a moverse en la clandestinidad. Surgieron los gérmenes de las organizaciones armadas que adquirirían protagonismo a partir del 70. En mayo del 69, en la Ciudad de Córdoba, el descontento popular produjo un importante levantamiento espontaneo de protesta contra la dictadura –llamado luego "el Cordobazo"-. Este levantamiento fue liderado por estudiantes y obreros y logró dominar importantes zonas de la ciudad durante varios días. El Cordobazo minó el sustento político de Onganía que al año siguiente recibiría su golpe de gracia con una acción del grupo armado justicialista "Montoneros": el secuestro seguido de muerte del Gral Aramburu. Pocos días después Onganía renunciaba asumiendo Levingston. A partir del 70 varios movimientos guerrilleros crecieron, apoyándose en el descontento popular: ERP, FAR, FAL como brazos armados de movimientos de izquierda y el ya citado Montoneros, justicialista.

4.3 Fundación de nuevas universidades nacionales.

La gravedad del Cordobazo exacerbó la preocupación de los conductores del régimen por el peligro latente que veían en la concentración de grandes masas de estudiantes. Esto dio origen a una política de atomización, que se cristalizó en la fundación de universidades nacionales en diversas ciudades del interior de país, destinadas a absorber al estudiantado local y quitar así masa crítica a las grandes universidades tradicionales. Así se fundaron entre otras las Universidades de Luján, San Luis, Río Cuarto, Tandil y La Pampa.

5. El breve período de democracia del Frente de Justicialista de Liberación Nacional.

En 1973 la situación de la "revolución Argentina" era insostenible. El movimiento Justicialista del General Perón, llevaba 18 años de proscripción cívica sin menguar ni el número de sus adherentes ni su gravitación política, el

[3] Una Vax 350, adquirida ya entrados los 80.

descontento popular se tornaba insostenible. Los movimientos guerrilleros extendían sus acciones y ganaban apoyo social. Finalmente el último presidente de la dictadura, el Gral. Lanusse, acordó una salida electoral sin proscribir al justicialismo por primera vez desde 1955, aunque impuso cláusulas legales que impedían la candidatura del mismo Perón. Las elecciones fueron ganadas ampliamente por el Frente Justicialista de Liberación Nacional (FREJULI), cuyo candidato era el Dr. Oscar Cámpora. En mayo de 1973, asumió Cámpora en medio de grandes festejos populares. En junio Perón regresó al país. Fue recibido por la mayor concentración popular de la historía argentina. Concurrieron al aeropuerto a recibirlo entre dos y cuatro millones de personas – según las distintas fuentes-. Sin embargo los festejos de ese día terminarían trágicamente. El ala derecha del movimiento justicialista disparó sobre sectores que exhibían y coreaban consignas revolucionarias. La concentración popular se tuvo que retirar bajo el silbar de las balas. Para facilitar el acceso de Perón a la presidencia Cámpora renunció y luego de nuevas elecciones asumió el Gral. Perón y luego, ante su fallecimiento en julio de 1975, su esposa Isabel Martínez.

El FREJULI inició una política de impulso de la industria nacional bajo la cual crecieron varios sectores industriales. En este contexto surgieron varios proyectos industriales para la construcción de equipamiento informático, entre ellos:

* Micro Sistemas, en Córdoba, que produjo estaciones de grabación de datos y micro computadoras usando microprocesadores estándar.

* Técnica Erova, en Buenos Aires, que a través de su subsidiaria Micro Computadoras Argentinas (MCA) produjo la MCA 3503, que contaba con un procesador Monroe de 8 bits y un impresor de agujas, bidireccional de diseño (del Ing. Carlos Bogni) y fabricación totalmente nacional, con la única excepción de la cabeza de impresión. Más tarde, ya durante la dictadura que seguiría a este período, MCA produciría la MCA 4503 con un procesador DEC de 16 bits (LSI-11 23).

* Fate Electrónica, que comenzó la fabricación de calculadoras, llamadas Cifra e inició el proyecto sin duda más importante: el diseño para su posterior fabricación de una línea de computadoras "Serie 1000". (El diseño de la Serie 1000 fue sumamente ambicioso. Apuntaba a una computadora ubicada en la frontera del Estado del Arte. Todo su software sería desarrollado en el país, el mismo se programaría en BCPL, un antecesor de c, grupos de Fate se ocuparían del sistema operativo, mientras que el desarrollo de las otras componentes del software de base fue licitado. El grupo de software del ONFCSM tendría a su cargo el desarrollo del compilador COBOL). En el contexto de este proyecto ingresó al país el primer UNIX, que corría en una PDP 11.

En la Universidad, durante el gobierno del FREJULI se pueden distinguir dos etapas bien diferenciadas:
La primera caracterizada por una gran movilización estudiantil y el predominio de criterios políticos en desmedro de los académicos, bajo control predominante del ala izquierda del justicialismo. Durante esta etapa se inició en la UBA el proyecto de ampliar los estudios de grado en Computación con una Licenciatura, que luego quedó trunco, concretándose recién en 1983.
La segunda, bajo control del ala derecha del FREJULI, comenzó con la intervención de Otalagano en la Universidad de Buenos Aires, en septiembre de 1974, extendiéndose a todas las universidades nacionales. Esta etapa se caracterizó por la persecución de los que habían participado en la anterior y de los adherentes al movimiento reformista y a la izquierda. Si algo se había construido en la primera etapa, fue destruido en la segunda.

Como consecuencia, durante este intervalo democrático no se produjeron avances en el estado de la enseñanza ni de la investigación de la Informática en el ámbito universitario argentino, tampoco en otras disciplinas. Así la Universidad no tuvo oportunidad de acompañar al proceso de desarrollo industrial, que por otra parte pronto quedaría trunco.

El período de gobierno del justicialismo fue sumamente tormentoso. los movimientos guerrilleros que inicialmente apoyaron al frente, luego volvieron a la clandestinidad. Apareció una fuerza para policial la "AAA", vinculada al ala derecha del gobierno y controlada por el Ministro del Interior, José Lopez Rega, que inició una extendida acción de intimidación y asesinatos, dirigida fundamentalmente al ala izquierda del mismo movimiento. El deterioro de la economía, el caos político y la violencia creciente fueron creando las condiciones que finalmente permitieron a los militares tomar nuevamente el poder.

6. La dictadura militar auto denominada "Proceso de Reconstrucción Nacional".

El 24 de marzo de 1976 las fuerzas armadas secuestraron a la presidente, Isabel Martínez de Perón y asumieron el gobierno. El Gral. Videla fue el primer presidente, luego lo seguirían Viola, Galtieri y Bignone. Los nuevos gobernantes instauraron una sangrienta dictadura que iba diezmar a la oposición, deteniendo a millares de jóvenes que, sin ningún amparo judicial y por razones muchas veces caprichosas, desaparecían, eran torturados y asesinados. Todas las dependencias del estado fueron intervenidas y los puestos claves ocupados por militares. Al poco tiempo de instaurado el gobierno, se dictó la "ley de prescindibilidad" que permitía dejar cesante discrecionalmente a cualquier empleado público. Su aplicación supuso la expulsión de todas las reparticiones y particularmente de las universidades e institutos de investigación de todos aquellos que no eran bien vistos por el nuevo poder y sus delegados. Entre otros, así se desmanteló al referido ONFCSM. Obviamente el sistema

académico pasó por un letargo que duraría lo que el régimen. Como siempre, hubo excepciones y algunos sectores aislados, fundamentalmente vinculados a temas de interés bélico, lograron continuar produciendo e incluso crecer.

La política monetaria de la dictadura, impulsada por su ministro Martínez de Hoz puso a la industria nacional en condiciones no competitivas, el mercado se llenó de productos importados y se produjo una gran retracción industrial, creciendo los sectores de servicios. Como consecuencia se truncaron los incipientes proyectos industriales informáticos antes mencionados. En cambio creció el parque de equipos de computación y se expandió el campo profesional. La demanda de profesionales informáticos produjo la apertura de nuevas carreras de grado como las de las Universidades de San Luis y Tandil. Estas se constituyeron en los centros académicos más activos de la época gracias al empuje de docentes locales como Raul Gallard y Jorge Boria en San Luis y Angel Orbe en Tandil. Empuje que buscó el apoyo de Profesores visitantes como Hugo Rickeboer y Armando Haeberer, Aguirre y el mismo Boria en Tandil.
Los años fueron debilitando al régimen. En 1982, el entonces presidente, Galtieri, buscó recuperar terreno con una gran acción que le permitiera ganar consenso y lanzó al país a la Guerra de Malvinas en contra de Gran Bretaña. Después del desastre bélico, de la ineptitud demostrada por los mandos a cargo de la acción y ante el creciente clamor popular y la hostilidad de las grandes potencias, que habían visto quebrar con la guerra el orden mundial que sustentaban, los militares se vieron obligados a llamar a elecciones y entregaron el poder.

7. El regreso a la democracia y el Programa Nacional de Informática y Electrónica.

En 1983 asumió el Dr. Raúl Alfonsín encabezando un amplio frente electoral, en nuevamente en medio de una gran euforia popular. Se recuperaba el sueño de lograr un desarrollo científico, tecnológico e industrial que disminuyera la brecha tecnológica, permitiendo una mayor independencia y bienestar. El nuevo gobierno para poder cumplir estas aspiraciones fijó entre sus prioridades el logro de un rápido desarrollo de la Informática (una descripción mas detallada de este proceso puede verse en un trabajo del autor y de Raúl Carnota [7] . Esta estrategia se basaba en la concepción – originada en los 70- de lo que se llamaba la Tercera Revolución Industrial: el nacimiento de un nuevo paradigma tecnológico económico ([7, 2]), que trasladaba el centro de interés de la economía hacia la tecnología de punta, especialmente la Informática (junto a la Electrónica), la Biotecnología y los Nuevos Materiales. La misma preocupación ya había llevado en los setenta a la Oficina Intergubernamental para la Informática (IBI), organismo de la ONU con sede en Roma, a promover lo que se dio en llamar "Políticas Nacionales de Informática" para los países no desarrollados, con el explícito objetivo de evitar que se profundizara la brecha tecnológica.

Alfonsín eligió para liderar el proceso de impulso de las nuevas ciencias y tecnologías al Dr. Sadosky, ya mencionado como fundador del Instituto de Cálculo e introductor de la computadora en el país. Sadosky fue puesto al frente de la Secretaría de Ciencia y Técnica (SECyT) que, recalcando su rol estratégico, dependería directamente de la Presidencia de la Nación. En 1984 la SECyT formó la Comisión Nacional de Informática, integrada por varias áreas de gobierno nacional y un miembro de las universidades nacionales (el Dr. Hugo Scolnik de la UBA). Entre las recomendaciones de esta Comisión surgió el Plan Nacional de Informática que sería implementado bajo control de la Subscretaría de Informática a cargo del Dr. Carlos Correa. El plan contemplaba estrategias tanto para el área industrial como para la académica.

Para la componente industrial de la estrategia se elaboró un ambicioso plan de fomento y protección. Plan que no pudo vencer ni los intereses de las compañías multinacionales ni los abusos de algunos empresarios locales y finalmente quedó en el olvido.

La componente académica resultaba fundamental dado que la investigación y desarrollo en Computación era un área de vacancia, mientras que la industria que se deseaba impulsar requeriría del apoyo de investigadores y tecnólogos para lograr su despegue. Para esta componente se concibieron dos proyectos troncales [2]: el Programa Argentino Brasileño de Investigación y estudios avanzados en Informática -PABI- en cooperación con Brasil y la Escuela Latino Americana Informática –ESLAI-, emprendimiento regional subsidiado fundamentalmente por el IBI. Estos proyectos tuvieron un importante impacto académico por lo cual sus características serán resumidas a continuación.

7.1 El Programa Argentino Brasileño de investigación y Estudios Avanzados en Informática (PABI).

Este programa se formalizó mediante un convenio firmado en 1986 por los gobiernos Argentino y Brasileño. Sus Coordinadores serían Carlos Pereira Lucena y Armando Haberer por Brasil y Argentina respectivamente. Comprendía la realización de proyectos con participantes de ambos países, el intercambio de investigadores y la realización periódica de encuentros de investigadores. También se programaba la realización conjunta de Escuelas para universitarios. El PABI realizó varios encuentros de investigadores y financió varios proyectos de investigación. Sin embargo, la carencia de grupos consolidados en la Argentina mostró que la primera etapa debía dedicarse fundamentalmente a la formación de recursos humanos. En consecuencia la realización de las Escuelas Argentino Brasileñas de Informática fueron su actividad más importante.

Las Escuelas Argentino Brasileñas de Informática (EBAI) .

Los objetivos de las EBAI eran contribuir a la conformación de una masa crítica de investigadores, a calificar la enseñanza de grado y a la creación de una escuela de pensamiento regional en la disciplina.
El diseño original preveía la realización de una Escuela por año. Se realizarían alternadamente en cada uno de los dos países. Habría un cupo de 250 alumnos por país cada uno de los cuales gozaría de una beca; 100 de ellos debían ser alumnos avanzados – cursantes del último año o recién graduados- mientras que los restantes debían haber completado al menos un segundo año de estudios universitarios. Las becas serían concursadas, realizándose la selección por antecedentes. Los cursos avanzados durarían una semana y los demás dos. Un característica importante fue la obligación de que cada profesor escribiera un libro sobre el tema de su curso, cuyo original debía presentar con antelación suficiente para que estuviera editado al comienzo de la EBAI correspondiente.
La primera EBAI se realizó en febrero de 1986 en Campinas. En la segunda (Tandil, 1987) además de los cursos se desarrollaron laboratorios de tres semanas. En ellos los participantes debían completar un proyecto, usando la tecnología de punta que en él se presentaba. Los primeros laboratorios fueron de Microelectrónica e Ingeniería de Software. El alumnado de la segunda EBAI tuvo la siguiente composición: 243 brasileños, 239 argentinos, 15 chilenos, 15 uruguayos, 6 peruanos, 3 cubanos y 2 bolivianos. Para la selección de esta Escuela se inició la modalidad de otorgar el 25% de las becas en forma automática a los alumnos que hubieran obtenido los mejores promedios en la anterior. Siguieron la III-EBAI en Curitiba, 1988 y la IV en Termas de Río Hondo 1989, con características similares. Ese año asumió la presidencia el Dr. Carlos Menem. El equipo de coordinación del PABI fue cambiado y la nueva conducción de la SECyT dejó de prestarle el decidido apoyo de su predecesora. Motivos presupuestarios impidieron continuar con la frecuencia anual. Posteriormente se realizarían, ahora dirigidas sólo a docentes e investigadores, la V-EBAI en 1991 en Nova Fiburgo y la VI-EBAI en Embalse Río Tercero en 1993. Luego de dos años de inactividad, el PABI desaparecería en 1995.

A partir de su inicio las EBAIs comenzaron a constituirse en el punto de encuentro e intercambio de las nuevas generaciones de estudiantes avanzados y recientes egresados. A la par, los Encuentros de Investigadores y algunas posibilidades de financiación comenzaron a forjar proyectos comunes de investigación y desarrollo. La interconexión de ambos eventos y su dinámica permitían pronosticar su efectividad para la creación de una masa crítica de investigadores y profesionales altamente capacitados, con visión latinoamericana. Sin embargo su pronto final no permitió establecer una tradición. Como resultado 2000 estudiantes recibieron cursos o participaron en laboratorios de los temas de punta en la disciplina y pudieron compartir un ámbito con los investigadores que en la época que trabajaban en esos temas. No menos trascendente, fue la colección EBAI de aproximadamente sesenta títulos originales en portugués y castellano. Esta colección fue distribuida a todos los centros de estudios superiores de la región en los cuales cumplió una importante función de difusión renovadora. Todavía a mediados de los 90 muchos textos se usaban en las carreras informáticas, algunos de los cuales debieron ser reeditados.

7.2 La Escuela Latino Americana de Informática (ESLAI).
La ESLAI debía constituir un centro de excelencia en docencia universitaria e investigación en Informática para la región de Latinoamérica y el Caribe. Mediante la ESLAI se pretendía elevar el nivel general de los sistemas regionales de formación profesional, superando el atraso del sector. Se esperaba lograr este objetivo mediante el proceso de difusión que supondría la inserción en el sistema académico y productivo de un grupo reducido pero altamente capacitado de graduados. Además a través de la ESLAI se aspiraba realizar una importante labor de capacitación profesional, brindando a la comunidad informática cursos de actualización, dictados por especialistas de primer nivel internacional. A mediano plazo, la ESLAI debía constituir una Escuela de Posgrado, cuando la reinserción de sus egresados, doctorados en el exterior, y la conformación de grupos propios de investigación, permitiera lograr masa crítica para ello. Al proyecto prestaron acuerdo varios países de la región y el IBI otorgó el principal financiamiento para su funcionamiento. La infraestructura fue provista por el Estado Argentino y por el Gobierno de la Provincia de Buenos Aires. Un importante equipamiento, para la época, fue donado por el Gobierno Italiano – una red constituida por dos servidores UNIX (AT&T 3B2) y 57 PC´s -. La UNESCO subsidió la compra de material bibliográfico. Convenios con Italia y Francia permitieron que seis profesores italianos y dos franceses dictaran sendas asignaturas durante un semestre cada uno. La Comunidad Económica Europea otorgó un primer subsidio de treinta mil dólares para la contratación de especialistas europeos para dictar cursos breves. Posteriormente, frente a los informes favorables presentados por los visitantes otorgó otro de setenta mil dólares y en el momento de su cierre se gestionaba un tercer subsidio de trescientos mil.
En gran medida estos apoyos se lograron gracias a la acción de un grupo de científicos argentinos, radicados en centros académicos del exterior, que apadrinaron la iniciativa (entre ellos Norma Lijtmaer, Mauricio Milchberg y Jacinto Aráoz). El complejo proceso de concretar el proyecto fue conducido entusiasta y eficazmente por Rebeca Guber y Armando Haeberer. A pocos meses de iniciadas las clases asumió la Dirección el Dr. Jorge Vidart.

Para lograr resultados rápidamente, aprovechando al máximo los recursos invertidos, se buscó minimizar el tiempo de permanencia de los estudiantes y garantizar tanto la mejor aptitud de los ingresantes como la dedicación exclusiva al estudio. Para esto se adoptó un esquema similar al que ya habían usado los físicos nucleares para fundar

el Instituto de Física de Bariloche, hoy "Instituto Balseiro" en honor a su principal promotor ([1]), fijándose los siguientes criterios:

1) Un ciclo de estudios en la Escuela de tres años. Este ciclo completaría una formación básica adquirida previamente aprobando al menos dos años de una carrera universitaria afín, cuya posesión debía probarse mediante la aprobación de la prueba de ingreso.
2) Un cupo de a lo sumo 35 alumnos seleccionados en una prueba tomada simultáneamente en todos los países de Latinoamérica y el Caribe. Prueba de ingreso que tenía el doble objetivo de seleccionar a los aspirantes más aptos y garantizar que los alumnos ingresaran con la formación previa asumida.
3) Una condición de dedicación plena al estudio por garantizada por medio de una beca de la que gozarían todos los alumnos.

La ESLAI funcionó con un cuerpo docente integrado por muy pocos Profesores Ordinarios, la mayoría de los cuales estaban contratados por períodos breves y por un conjunto también reducido de auxiliares docentes denominados Instructores. Los profesores eran seleccionados por un comité internacional y los Instructores por concurso.
El plan de estudio comprendía dos años de materias obligatorias y un tercer año donde se debía completar un determinado cupo de créditos de materias optativas. Además durante este último año debía realizarse una pasantía y la tesina de graduación. La primera era un trabajo de iniciación profesional y la última de iniciación en la investigación. Mediante las pasantías a realizarse en distintas empresas e instituciones se esperaba iniciar la vinculación de la ESLAI y sus egresados con el medio. El título – Licenciado en Informática – era otorgado, mediante un convenio especial, por la Universidad Nacional de Luján (La Universidad Nacional de San Juan, desempeñaba este rol en el caso del Instituto Balseiro).

La ESLAI inició sus clases en marzo de 1986. Los alumnos seleccionados en la primera cohorte provenían de distintas regiones de la Argentina, de Uruguay, Paraguay, Ecuador, Venezuela, Colombia, Perú.
Los cursos contaron con profesores de primera línea, entre los que puede citarse a Jean R. Abrial – asesor de la CEE -, Ugo Montanari – CNR -, Martin Virsing – Univ de Passau Alemania-, Carlo Ghezzi – Politécnico de Milán, Helmunt Partsch – Universidad Católica de Nijmegen -, Georgio Ausiello – Universidad de Roma - , Jean Pierre Jounnaud – Universidad de París-, Norma Lijtmaer – Univ de Pisa- .

Durante el periodo 1986 – 1989 los mismos profesores que dictaron los cursos del plan de estudios de la Escuela, (asignaturas semestrales de los dos primeros años y cursos optativos intensivos de distinta duración del tercer año), también dictaron 34 cursos abiertos a la comunidad, a los que asistieron 350 profesionales, docentes e investigadores.
El régimen de pasantías permitió iniciar una fructífera relación con el medio productivo, realizándose trabajos con importantes empresas e instituciones de Argentina, Brasil, Ecuador, Venezuela, Uruguay e Italia.[4]
Se constituyeron grupos de investigación y esta actividad, si bien incipiente, había producido, en 1990, 28 publicaciones y 54 presentaciones a congresos.
En 1988 se produjo la graduación de todos los egresados de la primera cohorte. Al año siguiente se produjo la segunda promoción. De los 59 alumnos que constituyeron estas dos primeras cohortes, habían egresado 54.
Algunos de los egresados retornaron a sus países y veinticinco egresados y cinco instructores partieron a realizar estudios de posgrado con becas provenientes del exterior – sin costos para sus países - a Inglaterra, Francia, Suecia, Holanda, Italia, Brasil, Alemania, Estados Unidos, Israel y Escocia.
Lamentablemente sólo estas dos cohortes, de las cuatro que alcanzaron a ingresar a la Escuela, pudieron completar su ciclo regularmente. La pérdida de interés en el proyecto en un sector crítico del gobierno que siguió al de Alfonsín, motivó que la ESLAI cerrara sus aulas en septiembre de 1990, pese al apoyo de toda la comunidad académica nacional, de numerosos empresarios y de importantes sectores de los dos partidos mayoritarios ([13]).

8. El regreso al neoliberalismo.

El gobierno del Dr. Alfonsín se enfrentó a la fuerza remanente de los sectores militares ligados a la dictadura anterior. En su primera etapa impulsó el juicio de las juntas de comandantes que habían detentado el poder por la desaparición de millares de personas. Luego tuvo que dictar leyes que daban impunidad a los mandos medios y subalternos tras varios golpes militares que nunca pudieron ser castigados por falta de apoyo de los sectores militares, que se decían leales pero no reprimían a los amotinados. Una primera corrida inflacionaria pudo ser contenida con el "Plan Austral" pero a fines de su período nuevamente se produjo una descontrolada hiper-inflación.

[4] ALUAR S.A (Argentina), Interfase S. A.(Uruguay), IdeaSoft (Uruguay), Petróleo de Venezuela S. A., Techint S.A., Propulsora Siderúrgica S.A.I.C(Argentina), SADE S.A (Argentina), SIDERCA S.A.(ARGENTINA), IBM Argentina S.A., SIGEBA S.A. (Argentina), TTI S.A: (Argentina), PRODAT (Argentina), Comisión Nacional de Energía Atómica (ARGENTINA), Corporación Estatal Petrolera Ecuatoriana, Facultad de Arquitectura, Facultad de Ingeniería, Facultad de Medicina y Facultad de Derecho de la Universidad de Buenos Aires, INCO Universidad de la República Oriental del Uruguay, Universidad de Pernambuco (Brasil), Universidad Federal de Río de Janeiro (Brasil), Universidad de Pisa (Italia).

En el 89 se realizaron elecciones de acuerdo al calendario constitucional. En ellas triunfó el Partido Justicialista cuyo candidato era el Dr. Carlos Menem. El gobierno del Partido Radical de Alfonsín debió adelantar el paso de mando, debido a una gran crisis económica, en la que se extendían los saqueos y el descontrol.

Menem, desde el comienzo de su gobierno, abandono las banderas populares que enarbolara en su campaña y la tradición antiimperialista del movimiento justicialista. Tras un primer período de inflación e inestabilidad económica puso al frente de la economía al Dr. Domingo Cavallo, quién impulsó la más decidida política neoliberal. Mediante la ley de convertibilidad el valor de un peso argentino quedó fijado en un dólar. Esta política cambiaria, llevó a aumentar considerablemente los costos internos, frente a los de los grandes países exportadores. En consecuencia favoreció cada vez más a la importación en desmedro de la producción nacional. Se produjo una total retracción de la industria nacional. Además las grandes empresas estatales fueron vendidas y junto con muchas empresas privadas pasaron a ser propiedad de capitales multinacionales. Cualquier intento americanista de paliar la dependencia de las grandes potencias quedó descartado, por las auto asumidas "relaciones carnales" del gobierno con Estados Unidos.
Los proyectos que causaban preocupación al Pentágono fueron neutralizados. El proyecto Cóndor de la Fuerza Area - construcción de un misil guiado de alcance continental- fue desmantelado. La Comisión de Energía Atómica, que había llegado a producir agua pesada y a vender tecnología nuclear, fue dividida y su planta de investigadores sufrió significativas mermas por un plan de retiro voluntario. Este plan, impulsado desde el gobierno, tentaba a retirarse a sus a sus investigadores, a los que por abandonar su puesto en la Comisión y sin que ello creara ninguna inhibición para trabajar en otra institución, se ofrecía una importante suma de dinero, que aumentaba con su antigüedad y rango.
Sin embargo, quizá buscando remedar al modelo científico americano, las universidades vieron algunas mejoras. En el 92 la situación salarial de los docentes exclusivos tuvo una mejora significativa y hacia el 94 -por iniciativa del titular de la Secretaría de Políticas Universitarias (SPU) del Ministerio de Educación Lic. Juan Carlos Del Bello- se implementaron dos proyectos nacionales destinados: Uno a propiciar la investigación en la Universidad, que pasaba a premiarse con un suplemento salarial (Sistema de Incentivos Docentes a la Investigación). El otro a mejorar la calidad de la enseñanza en el Sistema Universitario, el Fondo de Mejora de la Enseñanza de la Ciencias (FOMEC). El FOMEC tuvo una incidencia importante en el desarrollo de la Computación en la Universidad y se resumirá a continuación. La SPU también propició la instalación de la Red Inter Universitaria (RIU) que permitió la conexión a Internet de todas las universidades nacionales, hacia 1995.

8. 1 El Fondo para la Mejora de la Calidad en la Enseñanza de grado de las Ciencias (FOMEC).
El diseño del proyecto, coordinado por la Dra. Rebeca Guber, se inició en el 94 y se concretó en el 95. Para establecer un diagnóstico previo se realizó un relevamiento de la situación en que se encontraban las universidades nacionales en cada disciplina. Los resultados indican claramente el estado de atraso en que se encontraba la Informática:

Se relevaron 21 universidades nacionales con carreras de Informática. En total tenían casi 5000 alumnos. egresaba anualmente, aproximadamente el 3% de los que ingresaban en el mismo período. La planta docente global contaba con sólo dos doctores en informática, de los cuales uno tenía dedicación parcial. La disponibilidad de equipos era ínfima, llegándose, algunas universidades a un puesto de trabajo cada 50 alumnos. Lo mismo sucedía con las bibliotecas que disponían de 0.65 libros por alumno. Tres universidades habían iniciado carreras de doctorado con dirección externa (la UBA, la UNSL y la UNS), carreras que no tenían más de dos años de existencia y naturalmente, aún ningún graduado. En muy pocas universidades había grupos de investigación con producción.

Hacia fines del 95 el FOMEC quedó constituido. Se financiaba con un crédito del Banco Mundial al Gobierno Nacional. Tendría vigencia del 96 al 2000. Se otorgarían competitivamente fondos a proyectos presentados por unidades académicas de universidades nacionales. Dichas unidades podían ser desde universidades hasta cátedras. Las presentaciones debían contar con el aval de la universidad involucrada. El FOMEC pagaba un porcentaje de los montos y la universidad una contraparte que variaba según el rubro desde el 20 al 40 por ciento. Se disponía de un monto total de 250 millones de dólares (o pesos convertibles). Se realizaría una convocatoria anual, la primera se realizó en 1995 para comenzar a funcionar en 1996. Hacia 1999 se habían otorgado U$S 202.188.700 distribuidos por agrupación de disciplinas según lo muestra la figura 1 ([4]) (en el cuadro la Informática figura en Cs. Básicas).

Grupo de disciplinas	# Proy.	Mont	Grupo de disciplinas	# Proy.	Mont
Ciencias básicas	129	78.230	Ciencias Médicas	21	7.570
Ciencias Tecnológicas	114	56.908	Biblioteca	49	20.964
Ciencias Sociales	57	14.181	Desarrollo Institucional	54	10.697
Ciencias Humanas	48	13.636	Total	472	202.188

Figura 1.- Distribución de Proyectos FOMEC aprobados al 99 y Montos (en miles de dólares)

Los proyectos aprobados cubrieron, aunque con distintas proporciones a la totalidad de las 36 universidades nacionales.

Entre las 22 universidades con carreras de informática se habían otorgado 15 proyectos por un monto total de US$ 10.840.600. Los datos de la ejecución final de los proyectos de Informática, por universidad [4] se muestran en la figura 2 .

Los proyectos subsidiados por el FOMEC permitieron mejorar el equipamiento y dotar bibliotecas, pero fundamentalmente realizar una importante actividad de formación de recursos humanos. Muchos jóvenes salieron a cursar sus doctorados en el exterior, otros tuvieron becas para realizarlos en las carreras locales, fortalecidas por la posibilidad de traer profesores visitantes del exterior. Otros realizaron doctorados "sandwichs" con dirección compartida entre una universidad nacional y otra extranjera, realizando estadías de uno o dos años en el exterior. La enseñanza de grado contó también con el apoyo de profesores visitantes, en este caso en su mayoría provenientes de otra universidad argentina la de posgrado con profesores visitantes de universidades extranjeras. También se contemplaron pasantías de capacitación docente, que permitían que un docente permaneciera un período en una universidad más desarrollada en el aspecto de interés para capacitarse. Se constituyó una red de universidades con posgrados en Informática para compartir recursos permitiéndose que los cursos sirvieran para alumnos de otras universidades. Esta red, luego se extendió a cualquier universidad con carreras de Informática.

Universidad	Becas y pasnt. de posgrado	Equipamiento	Bibliografía	Consultores y Prof. Visitant.	Obras	Total
Buenos Aires	824,756.76	857,167.94	13,770.47	178,584.94		1,874,280.11
Centro de la Prov. De Buenos As.	218,013.45	330,827.74	43,847.77	71,348.86		664,037.82
Comahue	141,762.16	243,683.05	27,578.40	96,833.00		509,856.61
Córdoba	149,976.58	213,329.78	71,798.60	59,400.00		494,504.96
Entre Rios	94,642.50					94,642.50
La Matanza	196,460.23			28,907.98		225,368.21
La Plata	260,974.80	838,595.05	49,967.05	59,810.62	18,415.50	1,227,763.02
Patagonia Austral	92,729.95	126,735.20		32,725.48		252,190.63
Río Cuarto	45,741.55	76,637.03		141,358.18		263,736.76
Salta	63,794.43	110,565.13	8,008.63			182,368.19
San Luís	37,978.05	391,725.39	82,154.41	263,288.89		775,146.74
Sur	536,000.04	660,158.94	235,141.77	110,926.89	24,821.00	1,567,048.64
UTN Reg. Santa Fe	75,600.47	165,351.10	12,209.24	75,187.50	6,511.01	334,859.32
TOTAL	2,738,430.97	4,014,776.35	544,476.34	1,118,372.34	49,747.51	8,465,803.51

Figura 2.- Resultado de la ejecución de los proyectos FOMEC de Informática en dólares EE UU

9. Nacimiento de los estudios de posgrado.

En los años fundacionales de la Computación en la Universidad Argentina –1959 a 1966- algunas Universidades enviaron docentes a realizar posgrados en Ciencias de Computación al exterior, época en que sólo existían maestrías, (como la UNS que envió al Ing. Fontao a Stanford o la de Buenos Aires, que envió al Lic. Delbue también a Stanford). Luego hubo una gran despreocupación por la formación de posgrado. Ya durante el gobierno de Alfonsín, en el seno del SCyT en 1986 se diseñó un programa, por iniciativa del Dr. Jorge Santos, para lograr, que para 1997 se pudiera contar con 55 doctores y 50 magisters. El plan tendría un presupuesto de US$ 10 M., 6.7 M para becas y 3.3 M para equipar tres centros de excelencia. Los primeros 20 doctores realizarían sus estudios en el exterior. El programa no se llegó a implementar.

De la ESLAI partieron a realizar doctorados en Europa, Estados Unidos e Israel 23 graduados y 5 docentes auxiliares entre 1989 y 1990. También la Universidad de Tandil envió docentes a Brasil a realizar estudios de posgrado.

En 1992 la UBA abre, dentro de su Doctorado en Ciencias, la orientación Computación, bajo la dirección de Pablo Jacovkis. Poco después hacen lo mismo la Universidad Nacional de San Luis y la Universidad Nacional del Sur. En 1996 se defiende en San Luis, la primera Tesis Doctoral, es la de Turrul Torres con dirección de Alberto Mendelzon – Univ. de Toronto -. En 1997 en la UBA se doctora Martina Marré con la dirección de la Dra. Gallo –Univ. de Pisa-. A partir de allí, la producción de doctores se va acelerando y alcanza un ritmo sostenido, a la vez que comienza a haber direcciones locales.

10. La situación actual

El país cuenta ya con mas de 70 doctores y una importante cantidad de magisters. Esto se ha logrado gracias: a que muchos egresados de la ESLAI se insertaron en el sistema después de haber terminado sus estudios de doctorado en el exterior, al retorno de los que completaron sus becas FOMEC (o de otro origen) en el exterior y regresaron, a la producción de las carreras de posgrado locales.

Se ha roto el aislamiento de las universidades, cuyos docentes mantienen un contacto fluido. También, aunque más reducido existe contacto entre alumnos. A esto último contribuyen las Escuelas breves, de una o dos semanas. Al respecto el Departamento de Computación de la UBA realiza una de estas Escuelas (la ECI) durante el invierno desde hace más de quince años y la Universidad de Río Cuarto organiza la Escuela de Verano de Ciencias Informáticas todos los meses de febrero, desde 1994 . También los congresos suelen implementar escuelas.

Docentes de una importante cantidad de universidades se reúnen periódicamente para tratar de lograr un núcleo curricular común.
Existen grupos de investigación con muy buena producción e importantes conexiones internacionales.
Han comenzado a producirse ingresos en la Carrera del Investigador del CONICET de jóvenes investigadores informáticos y becarios.

11. Conclusiones

Las Ciencias de Computación han logrado crecer en la Argentina. Este crecimiento se ha producido pese a la falta de continuidad de las políticas académicas, a las frustraciones producidas por el truncamiento de proyectos exitosos, a los frecuentes períodos de asfixia económica del sistema científico. Este crecimiento ha sido logrado gracias a esfuerzos inteligentes y continuados de muchos universitarios que fueron acompañados por la comunidad y que supieron aprovechar los apoyos oficiales, cuando los hubo y superar las carencias cuando no.

El crecimiento de las de la Ciencias de Computación argentinas ha permitido a sus investigadores, doctorandos y proyectos ingresar al mayor organismo de promoción, el CONICET , cuyos estándares de evaluación los excluían hasta poco tiempo atrás.

Algunos de los proyectos que han ayudado al crecimiento mencionado han sido binacionales –PABI y EBAIs- o de alcance latinoamericano – ESLAI -. Estos proyectos no se habrían podido llevar a cabo dentro del marco exclusivamente nacional por falta masa crítica, y por la dificultad de conseguir financiamiento. El carácter regional ha facilitado el acceso al financiamiento de organismos internacionales.

Estos proyectos han tenido influencia importante en la Universidad uruguaya, algunos han sido aprovechados en Brasil y en su marco han recibido formación de grado de excelencia colombianos, ecuatorianos, peruanos, venezolanos y uruguayos.

Ahora resulta muy importante para la Universidad argentina retener dentro del sistema a una proporción importante de los recursos formados. Otro desafío es lograr una efectiva vinculación de la Universidad con el sistema productivo. Ambos temas pueden estar estrechamente vinculados y el desarrollo de sus vínculos quizás constituya la gran tarea futura.

Agradecimientos
El autor considera imprescindible agradecer: Al Dr. Manuel Sadosky, las enriquecedoras conversaciones mantenidas. A los doctores Jorge Santos, Carlos Abeledo y Nora Szasz, la información gentilmente brindada. Al Dr. Pablo Jacovkis, la entrega del manuscrito de su trabajo "Reflexiones sobre la historia de la Computación en la Argentina". Al Arq. Nicolás Babini, el amable obsequio de su último libro sobre la historia de la Computación argentina [3]. Finalmente, a Laura Pérez, sus agudas sugerencias y observaciones, y la paciente revisión del manuscrito de este trabajo.

Referencias

[1] J. A. Balseiro, crónica de una ilusión; A. Dávalos, N. Badino. Fondo de Cultura Económica, Argentina 1999
[2] Lineamientos de Política Científica y Tecnológica. SECyT-Argentina. 1984.
[3] La Computadora en la Argentina, crónica de una frustración. N. Babini. Editorial Dunken, 2003
[4] Documentos de la Dirección Ejecutiva del FOMEC. 2004
[5] El proyecto secreto Huemúl. Mario Mariscotti, Ed. Planeta, Argentina.
[6] La nuca de Houssay, M. Cereijido. Fondo de Cultura Económica, México 2000
[7] Dos emprendimientos regionales transformadores del sistema superior de enseñanza de Informática. J. Aguirre, R. Carnota. CLEI 2003, CIESC. La Paz, Bolivia 2003, pp 148.
[8] Cesar Milstein: Paradigma de la diáspora científica argentina, A. Khon Loncarica, N. Sánchez. Todo es Historia, Num 429 Buenos Aires, diciembre 2002 (Número especial) , pp 6-18.
[9] Investigación y difusión de la Física a comienzos del siglo XX, L. Andrini, C. Reichenbach. Todo es Historia, Num 429, Buenos Aires, diciembre 2002 (Número especial), pp 36-45.
[10] Entrevista a Manuel Sadosky. Un destino sudamericano, L. Moledo. Todo es Historia, Num 429 Buenos Aires, diciembre 2002 (Número especial), pp 46-50.
[11] Salvando la memoria de la computación en la Universidad de la República, Uruguay, a partir de los recuerdos del Prof. Manuel Sadosky, L. Bermúdez, M. E. Urquhart, UDELAR
[12] Breve resumen de la historia de la Computación, P. Jacovkis. Newsletter de Sadio, Num. 2, 2003. *www.sadio.org.ar*
[13] La Escuela Superior LatinoAmericana de Informática, advenimiento muerte prematura y proyección, J. Aguirre. Newsletter de SADIO Num 3, 2003. *www.sadio.org.ar*

ORGANIZAÇÃO CURRICULAR POR COMPETÊNCIAS EM CURSOS DE CIÊNCIA DA COMPUTAÇÃO INOVAÇÃO OU RECONTEXTUALIZAÇÃO?

Luiziana Rezende

Doutoranda pela Universidade Federal do Rio de Janeiro (UFRJ – COPPE)

Diretora do Departamento de Ciência da Computação e Informática da Universidade Gama Filho

Rio de Janeiro - Brasil

luiziana@ugf.br

Lídia Micaela Segre

Coordenadora de Pesquisa do Mestrado de Administração e Desenvolvimento Empresarial da Universidade Estácio de Sá e Profa. Colaboradora do Programa de Engenharia de Sistemas e Computação (UFRJ – COPPE)

Rio de Janeiro - Brasil

lídia@estacio.br

Gilda Helena B. Campos

Coordenação Central de Educação a Distância- PUC-Rio

Rio de Janeiro - Brasil

gilda@ccead.puc-rio.br

Resumo

Este artigo aborda as matrizes teórico-conceituais para organização curricular por competências em Cursos de Ciência da Computação, tentando identificar se este processo se constitui em inovação curricular ou em mera recontextualização que leve apenas ao eficientismo social e produtivo. Propõe também um método iterativo para identificação e mapeamento de competências a partir da análise de projetos pedagógicos de cursos de Ciência da Computação já implantados em Instituições de Ensino Superior. Este trabalho se inscreve no contexto de uma pesquisa mais ampla, ainda em andamento, sobre o desenvolvimento/mobilização de competências em cursos de Ciência da Computação no Brasil.

Palavras-chaves: Competências, Ciência da Computação, Matrizes teórico-conceituais, Currículo, Mapeamento de competências

1. INTRODUÇÃO

Atualmente, no Brasil, além do Currículo de Referência da Sociedade Brasileira de Computação (SBC), o documento mais utilizado pelas Instituições de Ensino Superior (IES) para construção de projetos pedagógicos na área de Computação e Informática é a Proposta de Diretrizes Curriculares de Cursos da Área de Computação e Informática [18], que ainda aguarda homologação pelo Ministério de Educação e Cultura (MEC). Também são utilizadas pelas IES as várias versões *do Computing Curricula da* ACM (*Association of Computing Machinery*) e do IEEE (*Institute of Eletric and Eletronic Engineering*) [1].

A nova Lei das Diretrizes e Bases da Educação no Brasil (LDB) introduz novos conceitos e filosofias para a educação superior no nível de graduação. De acordo com a LDB, a graduação é uma etapa inicial da formação e não um momento de esgotamento do conhecimento. Este aspecto dinâmico só é viável dentro de uma estrutura flexível, que permita aos cursos definirem diferentes perfis para os seus egressos, adaptando-os às rápidas mudanças do mundo moderno [3].

A Proposta de Diretrizes Curriculares de Cursos da Área de Computação e Informática [18] não indica claramente competências e habilidades a serem desenvolvidas nas modalidades de curso definidas para esta área (Bacharelados em Ciência da Computação, em Sistemas de Informação, em Engenharia da Computação; Licenciatura e Tecnologia), de forma a não criar "amarras" e não podar a criatividade das IES na construção de seus projetos pedagógicos, levando-se também em consideração a diversidade de sub-áreas específicas da área de Computação e Informática. Assim, é na proposição dos projetos pedagógicos que as IES definem as competências e habilidades que pretendem desenvolver, a partir do perfil do egresso definido na Proposta de Diretrizes Curriculares de Cursos da Área de Computação e Informática e dos objetivos do curso proposto.

Partindo da abordagem por competências, quais seriam, então, as estratégias, as proposições curriculares, os processos e os recursos no setor educacional, especificamente nos Cursos de Ciência da Computação, que deveriam ser mobilizados para alcançar uma formação sólida baseada na construção de competências? Como alcançar no desenvolvimento dos projetos pedagógicos pelas IES as competências necessárias para o pleno desenvolvimento de pessoas capacitadas aos desafios da sociedade e da profissão?

Este trabalho se inscreve no contexto de uma pesquisa mais ampla que investiga a mobilização e o desenvolvimento de competências em Cursos de Bacharelado em Ciência da Computação no Brasil, especificamente no Rio de Janeiro, a partir do levantamento, mapeamento e cruzamento de competências pretendidas nos projetos pedagógicos dos cursos analisados, assim como do perfil profissional de TI (Tecnologia da Informação) existente no mercado de trabalho regional e das competências requeridas para o pleno exercício do egresso na sociedade e em áreas específicas, como Desenvolvimento de Software, Banco de Dados e Redes de Computadores.

Este artigo aborda as matrizes teórico-conceituais para organização curricular por competências [11] em Cursos de Ciência da Computação, tentando identificar se este processo se constitui em inovação curricular ou em mera recontextualização que leve apenas ao eficientismo social e produtivo. Propõe também um método iterativo para identificação e mapeamento de competências a partir da análise de projetos pedagógicos de cursos de Ciência da Computação já implantados em Instituições de Ensino Superior.

Para tal, na segunda seção são apresentadas as dificuldades identificadas na construção da profissionalidade num currículo por competências na área de Computação; na terceira seção são abordadas as matrizes teórico-conceituais em que pode se pautar a organização curricular por competências [11]; na quarta seção é apresentado o processo de organização curricular por competências no Brasil e, finalmente, na quinta seção são abordados alguns métodos conhecidos para identificação e mapeamento de competências, como também são apresentados os princípios do Método Iterativo para Identificação e Mapeamento de Competências, desenvolvido e utilizado na pesquisa.

2. REQUISITOS PARA A CONSTRUÇÃO DA PROFISSIONALIDADE NUM CURRÍCULO POR COMPETÊNCIAS NA ÁREA DE COMPUTAÇÃO

"No início deste novo milênio, a Ciência da Computação tornou-se uma área extremamente vibrante e dinâmica" [1]. A computação vem se tornando uma área essencial no mundo contemporâneo. Os computadores estão integralmente presentes na cultura, no trabalho e na vida das pessoas. Além disso, o campo científico da

computação continua a evoluir num ritmo surpreendente. Novas tecnologias são introduzidas continuamente e aquelas já existentes tornam-se rapidamente obsoletas. Este fato tem um profundo efeito nos cursos de Ciência da Computação, afetando tanto o conteúdo quanto o processo pedagógico.

O avanço tecnológico na última década aumentou a importância de vários tópicos curriculares, entre outros: a *World Wide Web* e suas aplicações; tecnologias de rede, particularmente aquelas baseadas em TCP/IP; multimídia; banco de dados; interoperabilidade; programação orientada a objetos; a disseminação do uso de sofisticadas interfaces de programação de aplicações (APIs); interação humano-computador; segurança e criptografia [1].

No Brasil, o Curso de Ciência da Computação enquadra-se na categoria de cursos que têm a computação como atividade fim. Segundo a Proposta de Diretrizes Curriculares de Cursos da Área de Computação e Informática [18] os egressos do Curso de Ciência da Computação devem: a) estar situados no estado da arte da ciência e da tecnologia da computação, de tal forma que possam continuar suas atividades na pesquisa, promovendo o desenvolvimento científico, ou aplicando os conhecimentos científicos, promovendo o desenvolvimento tecnológico; b) estar capacitados no projeto e construção de *software* e no projeto de *hardware* e c) ser capazes de alavancar e/ou transformar o mercado de trabalho com idéias inovadoras.

Pode-se perceber claramente que no perfil do egresso do curso há ênfase na profissionalidade deste e em sua atuação/absorção no mercado de trabalho, além da preparação para a pesquisa e a atualização através de cursos de pós-graduação *lato e stricto-sensu*. A questão da competência profissional, então, é preponderante e permeia todo o processo de construção de projetos pedagógicos na área.

Na definição da competência profissional é necessário partir de uma visão global de profissionalidade do trabalhador, ou seja, de seu campo de responsabilidade, e em seguida verificar que ações e competências devem ser desenvolvidas e mobilizadas para que, nas melhores condições possíveis, esta possa ser assumida.

Para o desenvolvimento dessa profissionalidade que modelos de aprendizagem, que formações, que temporalidades, que acompanhamentos seriam necessários para que as competências realmente pudessem se desenvolver? Que relação as competências profissionais podem ter com os sistemas de formação formais e como podem ser reconhecidas e avaliadas? Essa é uma questão vital que vem sendo mundialmente analisada a nível educacional e organizacional.

O texto da proposta de Diretrizes Curriculares evidencia a preocupação com a aquisição e a construção de competências em consonância com o mundo do trabalho. Entretanto, como pode o sistema formal de ensino articular conhecimento e competências?

Alain Savoyant [23] propõe que "se distingam duas séries de questões para compreender o que está em jogo: a) como articular, na aprendizagem dos saberes profissionais, atividade em situação escolar e atividade em situação de trabalho quando a alternância só pode ser eficaz – e isso deveria ser evidente – quando se aprendem coisas semelhantes nas duas situações, mas por vias diferentes e complementares? b) Como inscrever a alternância em um quadro institucional de maneira tal que o indivíduo que está aprendendo – um jovem, por exemplo – seja realmente acompanhado, não fique isolado, deixado por sua própria conta?"

Assim, a alternância entre situação escolar e situação de trabalho engendra aprendizagens baseadas na experiência, as quais necessitam de orientação e acompanhamento pelo professor e não apenas aprendizagens baseadas na assimilação direta de conceitos.

Segundo Zarifian [29], "uma ação em situação escolar depende de uma atividade prática no sentido de que implica sempre a transformação, por meio de ações, de um objeto ou, mais comumente, de uma situação, ainda que o objeto em questão seja um conhecimento. Assim, a atividade, em situação escolar, é prática, no sentido pleno do termo, do mesmo modo que, ao contrário, a atividade em situação de trabalho mobiliza uma orientação intelectual. Logo, a separação entre teoria e prática não é, com certeza, uma separação entre situação escolar e situação de trabalho, mas, antes disso, ela é uma distinção interna a cada uma dessas situações".

É preciso que os saberes profissionais de referência sejam construídos, que a partir do trabalho profissional efetivo sejam construídos os conhecimentos e as representações próprias de cada campo de atividade, os conhecimentos e as representações que orientam e controlam o bom desenrolar da ação profissional. Essa explicitação dos saberes de referência que orientam e sustentam o trabalho de fato, raramente é realizada nas empresas.

Zarifian [29] aponta que "é preciso que os professores em situação escolar coloquem os estudantes nas práticas que engendram conhecimentos que se aproximam de um saber capaz de orientar uma ação profissional".

Alain Savoyant [29] propõe a "colocação em situação simulada" na qual é essencial que as atividades de orientação dos alunos pelo professor (e não apenas as atividades de execução) estejam o mais próximo possível das atividades utilizadas nas situações de trabalho reais.

Nestas situações, os conhecimentos profissionais de referência, construídos de maneira formalizada são importantes, sendo possível extrair deles as propriedades pertinentes a situação-problema que se procurará simular, para que se possa, então, deduzir os procedimentos de execução apropriados.

As questões aqui apresentadas constituem-se em requisitos para a construção de um currículo baseado na mobilização/desenvolvimento de competências para um Curso de Ciência da Computação, as quais deverão ser analisadas e estar previstas no desenvolvimento do projeto pedagógico do curso, para que este propicie aos alunos reais oportunidades de vivenciar a alternância entre situação escolar e situação de trabalho, preparando os egressos para acompanhar de forma crítica e autônoma a evolução tecnológica específica da área, assim como participar continuamente na construção da profissionalidade e da cidadania.

A próxima seção aborda as implicações da adoção de um modelo de competências para o currículo, caracterizando inicialmente as matrizes teórico-conceituais que orientam e embasam a organização curricular por competências e, finalmente, apontando alguns riscos possíveis na adoção de um currículo por competências dissociado de uma matriz teórico-conceitual de referência.

3. MATRIZES TEÓRICO-CONCEITUAIS PARA A ORGANIZAÇÃO CURRICULAR POR COMPETÊNCIAS

Um estudo introdutório realizado por Rezende, Segre e Campos [24], em cursos na Área de Computação e Informática no Brasil, que analisou alguns projetos pedagógicos de IES baseados em competências, apontou a carência do conceito de competência e do respectivo referencial teórico usado na organização curricular de todos os projetos analisados.

Várias são as concepções de competências existentes. Essas diferentes concepções, segundo Deluiz [11], "sinalizam para a existência de várias matrizes teórico-conceituais que orientam a identificação, definição e construção de competências, e direcionam a formulação e a organização do currículo". Estas matrizes estão fundamentadas em modelos epistemológicos e podem ser identificadas como: condutivista ou behaviorista, funcionalista, construtivista e crítico emancipatória, cada qual com uma análise própria do processo de trabalho com o propósito de identificação, definição e construção de competências profissionais.

As tabelas seguintes, construídas a partir do estudo de Deluiz [11], resumem essas quatro matrizes, na tentativa de uma visão comparativa sobre: a base teórico-conceitual e os fundamentos utilizados; os objetivos na análise do processo de trabalho com o propósito de identificação, definição e construção de competências profissionais; o conceito de competência usado; o objeto e o método de análise do processo de trabalho e a relação com o currículo.

Tabela 1 - Matriz Condutivista ou Behaviorista

MATRIZ CONDUTIVISTA OU BEHAVIORISTA	
Base	Psicologia de Skinner e Pedagogia dos Objetivos de Bloom.
Objetivo	➢ Identificar as tarefas de cada posto de trabalho e definir o currículo de formação; ➢ eficiência social.
Conceito de Competência	➢ Habilidade que reflete a capacidade da pessoa e descreve o que ela pode fazer e não, necessariamente, o que faz, independente da situação ou circunstância.
Objeto e Método de Análise	➢ O posto de trabalho e a tarefa para definir o currículo de formação; ➢ Análise ocupacional.
Relação com o Currículo	➢ Viés *behaviorista* relacionado à formulação dos objetivos de ensino em termos de condutas e práticas observáveis; ➢ Taxonomias intermináveis e fragmentação de objetivos; ➢ Currículo limitado, com estreita formação do trabalhador.

Tabela 2 - Matriz Funcionalista

MATRIZ FUNCIONALISTA	
Base	➢ Pensamento funcionalista na sociologia; ➢ fundamento metodológico-técnico e da Teoria dos Sistemas.

Objetivo	➢ Originar normas de competência de trabalho que descrevam resultados laborais a serem alcançados em uma área de trabalho determinada; ➢ eficiência da empresa/instituição.
Conceito de Competência	➢ Funções e tarefas especificadas nas normas de competência; ➢ subdivisão em unidades e elementos de competência.
Objeto e Método de Análise	➢ Identificação da função estratégica do setor ou da empresa e dos resultados esperados na atuação dos trabalhadores para que a função estratégica seja cumprida ➢ análise funcional.
Relação com o Currículo	➢ Currículo construído a partir das funções e tarefas especificadas nas normas de competência; ➢ aprendizagem se restringe às atividades e não aos seus fundamentos científico-tecnológicos; ➢ currículo limitado, com estreita formação do trabalhador.

Tabela 3 -Matriz Construtivista

MATRIZ CONSTRUTIVISTA	
Base	➢ Origem na França, sendo Bertrand Schwartz um de seus principais representantes.
Objetivo	➢ Identificar categorias para construção de um inventário de competências, em situações diferenciadas, de modo a se obter a compreensão da relação competência/contexto e seus processos de construção e evolução; ➢ categorias utilizadas na análise: cultura de base, conhecimentos científicos, técnicos e organizativos, saberes comportamentais e relacionais; ➢ constituir competências não só voltadas para o mercado, mas direcionadas aos objetivos e potencialidades do trabalhador.
Conceito de Competência	➢ Relação entre as atividades de trabalho e os conhecimentos incorporados e ou mobilizados; ➢ dimensão construtiva, processual, coletiva e contextual.
Objeto e Método de Análise	➢ Lista de habilidades e competências observáveis que o grupo de trabalhadores já possuía e/ou foram desenvolvidas durante o processo de formação/ação; ➢ busca do coletivo, tanto na análise do trabalho em suas relações com o contexto quanto da capacitação individual, compreendida dentro de uma capacitação coletiva.
Relação com o Currículo	➢ Possibilita a transposição das competências investigadas no processo de trabalho mediada por uma concepção pedagógica; ➢ a construção do conhecimento é um processo individual, subjetivo, de desenvolvimento de estruturas cognitivas, em uma perspectiva naturalista da aprendizagem, sem enfatizar o papel do contexto social para além da esfera do trabalho na aprendizagem dos sujeitos; ➢ baseada numa concepção ampliada de formação, mas que minimiza a dimensão sócio-política.

Tabela 4 -Matriz Crítico-Emancipatória

MATRIZ CRÍTICO-EMANCIPATÓRIA	
Base	➢ Ainda em construção, está baseada no pensamento dialético.
Objetivo	➢ Ressignificar a noção de competência, atribuindo-lhe um sentido que atenda aos interesses dos trabalhadores; ➢ apontar princípios orientadores para: investigação dos processos de trabalho, organização do currículo e proposta de educação profissional ampla.
Conceito de Competência	➢ Multidimensional, polissêmica, envolvendo facetas que vão do individual ao sócio-cultural, situacional (contextual-organizacional) e processual; ➢ construção balizada por parâmetros sócio-culturais e históricos.
Objeto e Método de Análise	➢ Investigação das competências no mundo do trabalho a partir dos que vivem as situações de trabalho, ou seja, dos próprios trabalhadores, identificando os seus saberes formais e informais, as suas formas de cultura e o patrimônio de recursos por eles acumulado (aprendizados multidimensionais, transferências, reutilizações) nas atividades de trabalho.

Relação com o Currículo	➢ Busca fazer a transposição das competências investigadas no processo e nas relações de trabalho, de modo a estabelecer no currículo o diálogo dos conhecimentos já formalizados nas disciplinas e a experiência do trabalho; ➢ a aprendizagem dos saberes disciplinares é acompanhada da aprendizagem dos saberes gerados nas atividades de trabalho: conhecimentos, valores, história e saberes da experiência; ➢ baseada numa dimensão social da construção do conhecimento articulada com a dimensão profissional e com a dimensão sócio-política.

Estas tabelas sintáticas nos auxiliam na identificação da matriz teórico-conceitual de referência em um projeto pedagógico de curso e na previsão de alguns impactos que sua adoção poderá causar.

Na construção de um currículo.a escolha sobre a concepção de competência adotada constitui-se numa tomada de decisão importante. Quando esta decisão está desvinculada da matriz teórico-conceitual subjacente a sua definição os resultados esperados podem não ser alcançados plenamente e haver muitas distorções entre o perfil do egresso desejado e o perfil do egresso alcançado.

Segundo Deluiz [11], "as escolhas em educação não são neutras e a escolha por um modelo de currículo baseado em competências vai expressar as características e os interesses dos grupos e das forças sociais que os elaboram".

Vários autores têm contribuído com indicações para a organização do currículo por competências Kuenzer [16], Deluiz [11], Zarifian [29], Perrenoud [22], Fleury [15], Tanguy [28], Naveira [20], mas se na construção dos projetos pedagógicos pelas IES estas não forem vinculadas as suas matrizes teórico-conceituais, estarão desprovidas de significado e poderão ter sua aplicação prática seriamente comprometida.

Ao relacionar o modelo de competência a sua matriz teórico-conceitual pode-se evitar repetir modelos educacionais reducionistas, que ora permanecem na tradição comportamentalista da origem do modelo de competências ou que ora favorecem somente a inserção dos egressos no mundo produtivo, com saberes necessários apenas a execução de atividades profissionais segundo às exigências de mercado (matriz condutivista e matriz funcionalista).

Além disso, é possível dissociar o currículo por competências de uma perspectiva não crítica de educação, sintonizada principalmente com os processos de inserção social e de controle de habilidades a serem desenvolvidas e, por conseguinte, de controle do trabalho docente e de favorecimento do eficientismo social (matriz condutivista e matriz funcionalista).

Outro perigo em que o desenvolvimento de um currículo por competências dissociado de sua matriz teórico-conceitual pode incorrer é se constituir na tentativa de responsabilizar os egressos pelo possível fracasso de sua inserção ou manutenção no setor produtivo, tornando-os responsáveis pelo desemprego, pelo subemprego ou pelo trabalho autônomo, como também pela exclusão, devido a sua incapacidade de adquirir/mobilizar as competências exigidas pelo mercado.

Ferreti [14] alerta para "[...] o modelo que trabalhe sobre o suposto de que tudo no campo profissional se torne responsabilidade individual". Araújo [2] complementa que " tal enfoque pode obscurecer o fato de que a definição, certificação e valorização das competências é uma questão política e histórica, uma vez que envolve interesses distintos e antagônicos entre capital e trabalho".

Assim, é importante observar e evitar que discursos curriculares conservadores e retrógrados surjam com nova aparência crítica e inovadora, recontextualizados, alcançando legitimidade, servindo essencialmente ao eficientismo social e produtivo.

Cada IES, ao organizar o currículo por competência, deve definir qual a matriz teórico-conceitual que deseja ter como base, atentando para a diferença entre as matrizes existentes e os efeitos que vão trazer ao currículo, tendo um mero caráter recontextualizador ou efetivamente um caráter inovador.

Ao abordar ainda as implicações da adoção de um modelo de competências para o currículo é importante ressaltar que "[...] é necessário refletir sobre a tradução das atividades listadas no perfil de uma determinada profissão para o currículo. A questão candente é: será que essa tradução está sendo realizada de maneira linear, mecânica, ou está sendo fruto de uma análise educacional mais ampla?" [13].

Desta forma, Depresbiteris [13] indica alguns aspectos que podem servir como reflexão na construção de currículos por competência para que não seja realizada uma tradução linear das atividades profissionais na construção dos projetos pedagógicos baseados em competências:

- Estruturação do conhecimento de acordo com um pensamento interdisciplinar;
- Desenvolvimento de capacidades que mobilizem as competências;
- Incentivo à resolução de problemas novos;
- Diversificação dos meios de desenvolvimento de competências;
- Contextualização do educando quanto à historicidade dos produtos de seu trabalho;
- Favorecimento de uma atitude de predisposição para com a profissão.

Podem ser agregadas a esta lista a abordagem multidiscplinar do conhecimento e a alternância entre situação escolar e situação de trabalho.

A reflexão sobre todos estes aspectos vai originar a inserção de componentes curriculares ou práticas pedagógicas na estrutura do curso, as quais serão fundamentais para mobilização/desenvolvimento de competências. Componentes curriculares ou práticas pedagógicas são elementos/recursos utilizados na estrutura curricular que vão estimular a mobilização e o desenvolvimento de competências. Como exemplos de componentes curriculares ou atividades pedagógicas podemos citar: aulas teóricas, aulas teórico-práticas, trabalho cooperativo, discussão dirigida, resolução de problemas, projetos multidisciplinares, monitoria, iniciação científica, estágio e projeto de fim de curso.

4. ORGANIZAÇÃO CURRICULAR POR COMPETÊNCIAS NO BRASIL

A última reforma do ensino no Brasil foi definida por uma legislação específica, a Lei de Diretrizes e Bases da Educação Nacional (LDB), aprovada em dezembro de 1996 e, num segundo momento, através dos pareceres, resoluções e portarias que surgem como forma de regulamentar a referida lei.

A LDB [3] traz em seu bojo a necessidade de estabelecer Diretrizes Curriculares para a educação básica e superior, substituindo o currículo mínimo, anteriormente utilizado, exceto na área de Computação e Informática. As Diretrizes Curriculares surgem como uma tentativa de dar maior flexibilidade e autonomia às IES, sendo de competência do MEC (Ministério de Educação e Cultura) e do Conselho Nacional da Educação a tarefa de orientar a organização curricular.

O processo de elaboração das Diretrizes Curriculares dos Cursos de Graduação no Brasil foi permeado por sugestões das diversas comissões de especialistas e, especificamente na área de Computação e Informática, foi amplamente discutido em listas de discussão da SBC pela Internet. Os princípios aspectos para as mudanças curriculares adotadas, desde então no Brasil, são: a) flexibilidade na organização curricular; b) dinamicidade do currículo; c) adaptação às demandas do mercado de trabalho; d) integração entre graduação e pós-graduação; e) ênfase na formação geral; f) definição e desenvolvimento de competências e habilidades gerais.

As Diretrizes Curriculares, então, "devem contemplar na sua elaboração a definição e o desenvolvimento de competências e habilidades, para os diferentes níveis de ensino, assim como o reconhecimento dos conhecimentos, habilidades e competências adquiridas fora do ambiente escolar" [3].

Entretanto, qual foi a matriz teórico-conceitual utilizada como base durante a construção das diretrizes curriculares nacionais para as diversas áreas? Esta matriz estava explícita e foi amplamente divulgada ou, ao contrário, estava implícita e ficou encoberta no bojo das discussões? Por que a decisão por uma abordagem por competências na educação nacional que "deve ser adotada nos diferentes níveis de ensino para tornar assim a educação formal mais próxima das experiências do mercado e da sociedade?" [7]. Esse conceito tem como significado a capacidade de articular, mobilizar e colocar em ação valores, conhecimentos e habilidades necessários para o desempenho eficiente e eficaz de atividades requeridas pela natureza do trabalho.

Que fatores motivaram relacionar a educação nacional diretamente ao mundo do trabalho, através da abordagem por competências? E, seguindo a abordagem por competências, qual seria então o conceito de competência utilizado?

Pelos documentos oficiais a definição de competência [7] é "a capacidade de mobilizar múltiplos recursos numa mesma situação, entre os quais os conhecimentos adquiridos na reflexão sobre as questões pedagógicas e aqueles construídos na vida profissional e pessoal, para responder às diferentes demandas das situações de trabalho".

Pode-se perceber, então, a estreita relação entre a abordagem por competências com o mundo do trabalho evidenciada nos textos oficiais sobre a educação nacional, assim como o direcionamento para a vivência de situações que requeiram a mobilização de competências não somente técnico-cognitivas, mas também técnico metodológicas no contexto educativo e nas práticas educacionais, além das pessoais e sociais [20].

Maués, Wondje e Gauthier [17] consideram que "o modelo de competências aparece na educação brasileira como uma senha que vai permitir a entrada no mundo contemporâneo e o alinhamento com a tendência internacional no que se refere à educação".

Desta forma, no Brasil, a inserção do modelo de competências na organização curricular tende a um caráter pragmatista, utilitarista, imediatista e de adaptabilidade à realidade do contexto social, ou seja, seguindo a lógica do mercado de trabalho, exigindo resultados e eficiência que sejam demonstrados a partir de atividades bem definidas e de desempenhos traduzidos em ações específicas, enfim, calcada na matriz teórico-conceitual do funcionalismo. Essa tendência pragmatista e imediatista poderá ter seu curso desviado a partir de reflexões profundas sobre as matrizes teórico-conceituais na organização curricular por competências e dos reais objetivos educacionais e de cidadania que se deseja alcançar.

Segundo Popkewitz [23] "[...] na realidade, o ensino por competências parece responder à globalização em curso, à nova ordem econômica mundial, visando criar uma certa hegemonia ideológica, facilitando, através do

ensino, a chamada revolução conservadora. Em verdade trata-se mais de um instrumento para realizar a regulação social".

Maués, Wondje e Gauthier [17] consideram que o modelo de competências no Brasil tem sua ênfase acentuada numa lista de desempenhos, isto é, no saber fazer, o que pode tornar a educação a partir dessa abordagem meramente técnica, instrumental e aplicada, e isso pode ser muito reducionista.

A introdução da noção de competências na organização curricular no Brasil poderá ocasionar várias mudanças em relação à organização curricular anterior centrada em objetivos. Somente dar ênfase às novas experiências e formas de trabalho em situação sem fornecer os saberes formais correspondentes pode levar ao fracasso. Entretanto somente a assimilação dos saberes formalizados não é suficiente para pressupor ações eficazes.

Cabe assinalar, que a introdução da noção de competências na organização curricular também poderá trazer repercussão na formação profissional do docente que, dependendo do modelo de base, deverá assumir um papel de pesquisador em ação, envolvido nas práticas por ele analisadas e tentando compreendê-las em toda a sua complexidade a fim de intervir para melhorá-las.

Assim, a realização de estudos empíricos, qualitativos, de casos, oriundos de outras pesquisas mais amplas ou a partir da seleção de dados colhidos em situações de formação, ensino e/ou avaliação devem ser realizados no contexto brasileiro, a fim de verificar a condução do processo de organização curricular por competências pelas IES, identificar as condições de sua implementação e os casos de sucesso e de fracasso, assim como o potencial e as limitações do modelo e da matriz teórico-conceitual utilizados.

Faz-se necessário nesses estudos a utilização de métodos para identificação e mapeamento de competências, tanto no âmbito educacional quanto no setor produtivo. A próxima seção aborda alguns métodos conhecidos para identificação e mapeamento de competências e apresenta os princípios do método a ser desenvolvido nesta pesquisa.

5. MÉTODOS PARA IDENTIFICAÇÃO E MAPEAMENTO DE COMPETÊNCIAS

Os processos e métodos para identificação e mapeamento de competências surgiram no contexto empresarial, sendo utilizados pelos setores de recursos humanos na tentativa de identificar as competências dos trabalhadores e mapeá-las por áreas ou níveis, com objetivos da gestão de pessoas.

Para tal foram criados instrumentos que explicitam as necessidades de desenvolvimento dos profissionais, a partir da comparação entre o nível de competências requerido e o demonstrado, tomando como base o que as pessoas entregam à organização, segundo Oliveira Neto [21]. O conceito de competência é visto "não apenas como o estoque de conhecimentos, habilidades e atitudes de um indivíduo, mas também os resultados, a produção e a entrega decorrentes de sua mobilização em situações de trabalho" [21]. Sob este enfoque são definidos diferentes níveis de complexidade de entrega para as competências, podendo ser agrupadas segundo eixos, tais como tecnológico, gerencial/administrativo e outros.

Oliveira Neto [21] apresenta também o mapeamento de competências a partir da Teoria da Abrangência [21], segundo a qual qualquer organização é uma realidade sistêmica que tem três tipos de processos: funcional, mental e motivacional, nos quais os funcionários sempre atuam. Assim, as competências devem ser analisadas levando-se em consideração estes três processos.

Os funcionários são analisados quanto às suas competências e de acordo com a função que ocupam no processo produtivo da empresa. Três subsistemas ou redes de fluxos de informação (rede material, rede intelectual e rede institucional) se responsabilizam pela junção dos processos que ocorrem na sua respectiva área, pela gestão destes processos, pelos contatos com o mundo exterior e pela forma e conteúdo da organização e do produto.

Além de se avaliar o nível de entrega nas competências escolhidas, pode-se, também, utilizar um instrumento para analisar o potencial energético dos funcionários nestas competências, que consiste numa ferramenta de auto-conhecimento que permite uma visão global tanto de cada competência, compreendendo pontos fortes ou fracos, quanto do movimento do processo de seu potencial energético profissional.

Rossato [25] apresenta um método para definir e mapear as competências dos colaboradores de uma empresa, sendo esta uma das etapas da Gestão do Conhecimento.

"A Metodologia da Gestão do Conhecimento é um conjunto de etapas que visa continuamente, analisar as características da organização, avaliar suas capacidades, potencialidades, oportunidades, ameaças, limitações ou distorções, identificar as turbulências e incertezas do mercado e as necessidades dos clientes, criticar os pressupostos ou as implicações da utilização do modelo de gestão, gerar ou experimentar novos métodos relacionados com o uso do Modelo e conduzir todo o processo de gestão do conhecimento, de modo que os elementos fundamentais da teoria do conhecimento sejam implantados com sucesso" [25].

Na fase de definição das competências dos colaboradores devem ser mapeadas as competências de cada colaborador e identificadas aquelas que são fundamentais para garantir a eficiência e a eficácia dos seus

processos de negócios. Esse mapeamento envolve tanto as competências emocionais, quanto as técnicas e acadêmicas.

No cenário empresarial, segundo Rossato [25] "a competência é uma habilidade portátil que permite desempenhar atividades e ações, adotar um comportamento, tomar decisões e atitudes, assumir responsabilidades, debater sobre um assunto e gerar resultados, podendo ser utilizada em diferentes contextos profissionais ou dentro dos objetivos organizacionais, contribuindo muito para o alto desempenho dos indivíduos, das equipes e da própria organização. Essa competência pode ser trabalhada, desenvolvida e melhorada, continuamente, ao longo da sua vida, à medida que as pessoas acumulam experiências, aprendem a controlar suas emoções e seus impulsos e conseguem se automotivar. Para tanto, é necessário que cada ser humano conheça suas próprias competências".

Mitchell, Fuks e Lucena [19] consideram que "[...] atualmente há padrões de mercado para a modelagem computacional de competências. Os mais difundidos são os propostos pelo IMS, pelo IEEE e pelo HR-XML. O primeiro inclina-se para o foco das instituições de ensino, mas por ser genérico, minimalista e extensível, pode ser usado somente como ponto de partida para os outros enfoques. O segundo é voltado para o conteúdo (objetos de aprendizagem). E o terceiro, para a gestão de recursos humanos".

Setzer [27] apresenta um método para construir matrizes de competência através do qual associa áreas de conhecimento a habilidades aplicáveis pelos profissionais. Um profissional pode ter várias matrizes de competência, dependendo da área analisada. O objetivo é quantificar quantos profissionais detêm uma certa competência mínima, ou acima dela, para que possam ser alocados em projetos específicos.

Grande parte da literatura revisada sobre identificação e mapeamento de competências está relacionada a métodos para identificar competências e habilidades de pessoas ou colaboradores de uma empresa ou em projetos contratados, como também em relacioná-las a conhecimentos em áreas específicas, geralmente usados pelo setor de Recursos Humanos das empresas.

Em relação a área educacional não foram encontrados métodos para identificação e mapeamento de competências requeridas para a organização curricular. A literatura encontrada registra apenas princípios básicos para uma pedagogia baseada em competências [9], listagem de competências e habilidades a serem desenvolvidas no ensino profissional superior, por área de conhecimento [8], uma metodologia para o estabelecimento de perfis profissionais para certificação profissional baseada em competências [5] e um conjunto básico de habilidades agrupadas em diversas competências básicas, as quais formam a base para uma educação orientada para a competência de crianças e jovens até cerca de dezoito anos de idade [10].

Desta forma, para esta pesquisa será desenvolvido um método próprio para identificação e mapeamento das competências nos projetos pedagógicos das IES e empresas, cujos princípios são apresentados a seguir.

5.1 Método Iterativo para Identificação e Mapeamento de Competências

A pesquisa na qual este artigo se inscreve tem entre seus objetivos identificar, extrair e mapear competências em projetos pedagógicos de cursos de Ciência da Computação, o que para ser alcançado carece de um método específico, ainda não encontrado na literatura revisada.

Devido ao seu caráter inovador, o processo de identificação e mapeamento de competências no contexto educacional deve contemplar a participação de professores e alunos, que precisa ser contínua, constituindo-se num processo iterativo, ou seja, num processo contínuo, que retroage às etapas anteriores para sofrer sucessivos refinamentos até se obter os mapas de competências das IES e das empresas.

O Método Iterativo proposto para Identificação e Mapeamento de Competência consiste de três etapas, baseadas nas atividades descritas a seguir:

1. Construção do mapa de competência pretendidas pelas IES:

Nesta fase são identificadas, classificadas e mapeadas as competências contidas no projeto pedagógico da IES, através da atribuição de pesos e da associação das disciplinas e dos componentes curriculares às competências que mobilizam/desenvolvem, através das seguintes etapas**:**

- Identificação de competências no projeto pedagógico da IES;
- Mapeamento das competências por tipo: pessoais ou subjetivas, sociais ou comunicativas, naturais ou cognitivas: técnicas ou metodológicas;
- Atribuição de pesos de 1 a 3 para cada competência, de acordo com o grau de preponderância no desenvolvimento do currículo do Curso de Ciência da Computação;
- Associação das competências às disciplinas e aos componentes curriculares mobilizados durante o desenvolvimento do curso.

2. Construção do mapa de competências requeridas pelas empresas, por áreas de conhecimento:

Nesta fase são identificadas, classificadas e mapeadas as competências requeridas dos profissionais nas empresas e organizações, através da atribuição de pesos e da associação com as áreas de conhecimento através das seguintes etapas**:**

- Identificação de competências requeridas pela empresa;
- Mapeamento das competências por tipo: pessoais ou subjetivas, sociais ou comunicativas, naturais ou cognitivas: técnicas ou metodológicas;
- Atribuição de pesos de 1 a 3 para cada competência, de acordo com o grau de preponderância do profissional na área de conhecimento relacionada;

3. Cruzamento dos mapas de competências da IES e da empresa com o projeto pedagógico.

Ao final dos processos anteriores foram obtidos dois mapas de competências, da IES e da empresa, que ao serem cruzados com o projeto pedagógico da IES nortearão as análises necessárias.

Através do mapeamento de competências será possível detectar inconsistências nos projetos pedagógicos das IES analisadas, avaliar a adequação da carga horária das disciplinas aos tipos de competência que se pretende mobilizar/desenvolver durante todo o curso, verificar quais são os componentes curriculares mais adequados e verificar com que incidências devem ser utilizados, verificar se as competências desenvolvidas na IES são realmente as competências requeridas no mercado de trabalho, enfim, analisar criticamente o projeto pedagógico a partir de uma matriz teórico-conceitual e das necessidades reais do mercado de trabalho.

Nesta fase os mapas de competências da IES e das empresas são cruzados com o projeto pedagógico da IES, de forma a se avaliar se os objetivos previstos e as competências pretendidas poderão ser realmente alcançados e quais as disciplinas e os componentes curriculares essenciais para seu alcance.

O processo iterativo pode ser visualizado de forma mais clara através do gráfico (Fig. 5.1.1) seguinte, assim como um exemplo de mapeamento de competências de uma IES.(Fig. 5.1.2).

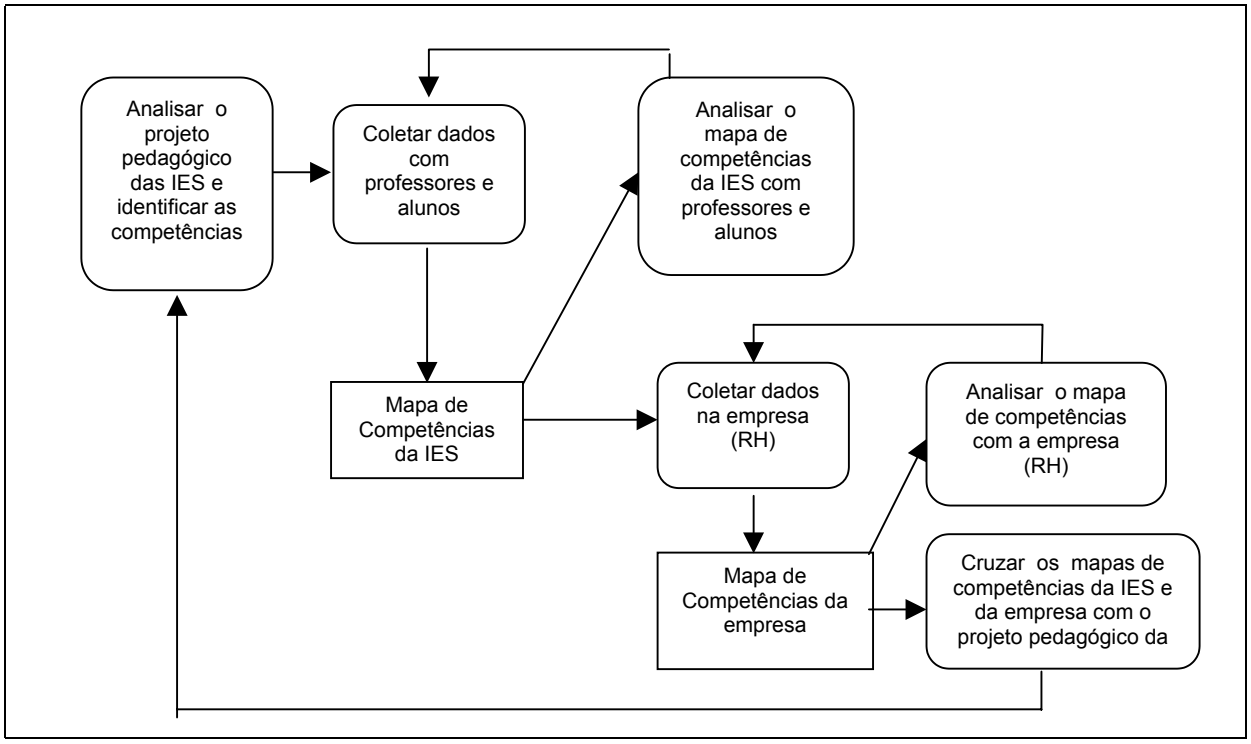


Fig. 5.1.1 – Processo Iterativo para Identificação e Mapeamento de Competências

TIPO	COMPETÊNCIA	PESO	DISCIPLINAS					COMPONENTES CURRICULARES				
4	Resolver eficientemente problemas em ambiente computacional.	3	2	21	3	18	12	1	2	3	4	5
			24	13				6	7	8	9	10
1	Ter espírito crítico e comportamento ético.	3	10	9	26			1	2	3	4	5
								6	7	8	9	10

3	Criar, implantar e desenvolver processos de software.	3	3	12	2	6	13	1	2	3	4	5
			16	18	26			6	7	8	9	10
3	Prestar serviços de consultoria ou assessoria a empresas de diversas áreas quanto ao uso adequado de sistemas computacionais.	2	1	3	6	9	10	1	2	3	4	5
			12	17	25	26		6	7	8	9	10
2	Participar, de forma colaborativa e integrada, em equipes que desenvolvem projetos na área de informática.	2	1	3	9	12	25	1	2	3	4	5
			26	11				6	7	8	9	10
1	Descobrir e empreender novas oportunidades para aplicação de sistemas computacionais e avaliar a conveniência de investimentos e desenvolvimento da aplicação.	2	11	3	12	9	10	1	2	3	4	5
								6	7	8	9	10

TIPO DE COMPETÊNCIA	PESO DA COMPETÊNCIA	COMPONENTES CURRICULARES			
1. Pessoal	3. Relevante	1	aulas teóricas	6	projetos multidisciplinares
2. Social	2. Pouco Relevante	2	aulas teórico-práticas	7	monitoria
3. Cognitiva Técnica	1. Irrelevante	3	trabalho cooperativo	8	iniciação científica
4.Cognitiva Metodológica		4	discussão dirigida	9	estágio
		5	resolução de problemas	10	projeto de fim de curso

DISCIPLINAS		
1. Administração de Empresas	10. Direito e Ética em Informática	19. Linguagens Formais
2. Algoritmos	11. Empreendedorismo	20. Língua Portuguesa
3. Análise e Projeto de Sistemas OO	12. Engenharia de Software	21. Lógica Matemática
4. Complexidade de Algoritmos	13. Estrutura de Dados	22. Matemática Combinatória
5. Computação Gráfica	14. Física	23. Matemática Discreta
6. Banco de Dados	15. Inglês Técnico	24. Probabilidade e Estatística
7.Cálculo Diferencial e Integral	16. Inteligência Artificial	25. Psicologia
8. Cálculo Numérico	17.Interface Humano-Computador	26. Sociologia
9. Computador e Sociedade	18. Linguagem de Programação OO	27. Teoria Computacional

Fig. 5.1.2 Exemplo de Mapa de Competências de uma IES

6. Conclusão

Várias IES vêm construindo projetos de curso baseadas no modelo de competências, sem contudo definir qual é esse modelo e qual sua matriz teórico-conceitual de base, o que poderá trazer distorções no perfil do egresso que se deseja formar. Há, portanto, necessidade de se ter conhecimento e reflexão sobre essas matrizes teórico-conceituais e as implicações de sua adoção na base do desenvolvimento curricular de cursos, especificamente na área de Computação, objeto de estudo desta pesquisa.

Existem vários métodos de mapeamento de competências na área de Recursos Humanos, que pouco se aplicam no mapeamento de competências em currículos para cursos de Computação. O método proposto neste trabalho está sendo validado em estudos de caso realizados em quatro IES do município do Rio de Janeiro e tem

o propósito de gerar debates e contribuições que auxiliem na melhoria da construção curricular baseada em competências na área de Computação.

Através do mapeamento de competências será possível detectar inconsistências nos projetos pedagógicos das IES analisadas, adequar a carga horária das disciplinas aos tipos de competência que se pretende mobilizar/desenvolver durante todo o curso, verificar quais são os componentes curriculares mais adequados e com que incidências devem ser utilizados, verificar se as competências desenvolvidas na IES são realmente as competências requeridas no mercado de trabalho, enfim, analisar criticamente o projeto pedagógico a partir de uma matriz teórico-conceitual e das necessidades reais do mercado de trabalho.

De forma geral, os resultados da pesquisa permitirão revelar processos, métodos e recursos para o desenvolvimento e mobilização de competências na formação de profissionais na área de Tecnologia da Informação, em consonância com a expectativa do mercado de trabalho e da sociedade como um todo. Os resultados também servirão como um indicador para análises futuras do perfil do egresso do Curso de Ciência da Computação proposto nas Diretrizes Curriculares e utilizado em grande parte dos currículos dos cursos de graduação no Brasil.

REFERÊNCIAS BIBLIOGRÁFICAS

[1]ACM/IEEE. Proposta de currículo. 2001. Disponível em: www.computer.org/education/cc2002/ironman/cc2001/index.html. Acesso em 17 set 2002.

[2] ARAÚJO, R. M. de L. A reforma da educação profissional sob a ótica da noção de competências. Boletim Técnico do SENAC. Rio de Janeiro: v.28, n.3. set/dez., 2002.

[3]BRASIL. Lei de Diretrizes e Bases da Educação, Lei no. 9394, de 20 de dezembro de 1996.

[4]BRASIL. Certificação de competências profissionais: discussões. Brasília: OIT, TEM/FAT, 1999.

[5]BRASIL. Metodologia para o estabelecimento de perfis profissionais. Projeto Estratégico Nacional. "Certificação Profissional Baseada em Competências. Brasília: 2000.

[6]BRASIL. SEMTEC. Ministério da Educação e Secretaria de Educação Média e Tecnológica. Educação profissional: referências curriculares nacionais da educação profissional de nível superior. Brasília, 2001.

[7]BRASIL/CNE. Resolução CEB. n.4, de 8 de dezembro de 1999. Institui as Diretrizes Curriculares Nacionais para a Formação de Professores de Educação Básica, em Nível Superior, Curso de Licenciatura e de Graduação Plena.

[8]BRASIL/CNE. Parecer n. 009/2001. Institui as Diretrizes Curriculares Nacionais para a Educação Profissional de Nível Técnico.

[9]BURNIER, S. Pedagogia das competências: conteúdos e métodos. Disponível em: <www.ilo.org/public/spanish/region.> Acesso em: 26 fev 2004.

[10]CHAVES, E. O. C. Educação orientada para competências e currículo centrado em problemas. Disponível via URL http://chaves.com.br, 2004.

[11]DELUIZ, N. O modelo das competências profissionais no mundo do trabalho e na educação: implicações para o currículo. Boletim Técnico do SENAC. Rio de Janeiro: v.27, n.3. set/dez., 2001.

[12]_____. Formação do trabalhador: produtividade e cidadania. Rio de Janeiro: Shape, 1995.

[13]DEPRESBITERIS, L. Analisando competências na escola de alguns ou na escola de todos? Boletim Técnico do SENAC. Rio de Janeiro: v.27, n.3. set/dez., 2001.

[14]FERRETI, C. J. Comentários sobre o documento Diretrizes Curriculares para a Educação Profissional de nível técnico MEC/CNE. [s.:l: s.n.] 1999. texto.

[15]FLEURY, A. e FLEURY, M.T.L. Estratégias empresariais e formação de competências. Atlas: São Paulo, 2001.

[16]KUENZER, A. Z. Conhecimento e competências no trabalho e na escola. Boletim Técnico do Senac, Rio de Janeiro, v.28, n.2, maio/ago., 2002.

[17]MAUÉS, O. C.; WONDJE, C.; GAUTHIER, C.Duas perspectivas diferentes em relação à abordagem por competências no ensino: os casos do Brasil e do Quebec. GT de Formação de Professores, n. 08. Disponível em: www.anped.org.br/25/olgaisesmauest08.rtf. Acesso em 25 fev 2004.

[18]MEC/SESu/CEEInf. Proposta de diretrizes curriculares de cursos da área de Computação e Informática. Brasília: Disponível via URL: www.mec.gov.br/sesu. 1999.

[19]MITCHELL, L. H. R. G.; FUKS, H.; LUCENA, C. J. P. Extensão de modelos de competências para avaliação formativa e continuada e planejamento de recursos humanos. SBIE 2003.

[20]NAVEIRA, R. B. Formação profissional para o século XXI. Rio de Janeiro: Atlas, 1995.

[21]OLIVEIRA NETO, , L. A. Mapeamento de competências: o enfoque da teoria da abrangência. 2002. Disponível em <projeto.org.Br/kmbrasil/Leopoldo.ppt>. Acesso em 05 fev 2004.

[22]PERRENOUD, P. 10 novas competências para ensinar. São Paulo: Artmed, 2002.

[23]POPKEWITZ, T. S. Reforma educacional, uma política sociológica. Porto Alegre: Artmed, 1997.

[24]REZENDE, L.; SEGRE, L. M.; CAMPOS, G. H. B. O projeto de fim de curso como componente curricular para o desenvolvimento e mobilização de competências nos cursos de graduação na área de computação e informática: um estudo introdutório. Anais do XXIII Congresso da Sociedade Brasileira de Computação (WEI). São Paulo, Campinas: agosto 2003.

[25]ROSSATO, M.A. Gestão do conhecimento: a busca da humanização, transparência, socialização e valorização do intangível. Rio de Janeiro: Interciência, 2002.

[26]SBC. Currículo de referência (CR). Disponível em <www.sbc.org.br >. Acesso em 15/08/2003.

[27]SETZER, V. W. Dado, informação, conhecimento e competência. Disponível em: <www.ime.usp.br/~vwsetzer/dado.info.html>. Acesso em: 29 fev 2004.

[28]TANGUY, L. **Saberes e competências:** o uso de tais noções na escola e na empresa. São Paulo: Papirus, 1997. p. 25-68.

[29]ZARIFIAN, P. Objetivo competência: por uma nova lógica. Tradução Maria Helena C. V. Trylinski. São Paulo: Atlas, 2001.

Que tipo de profissionais estamos formando? Relato de uma experiência

Gentil J. de Lucena Filho

Universidade Católica de Brasília , Mestrado em Gestão do Conhecimento e da Tecnologia da Informação

Brasília – DF, Brasil, CEP: 70790-160

gentil@pos.ucb.br

Margarita M. Morales Villegas

Construindo Juntos – Coaching, Consultoria e Treinamento

Brasília – DF, Brasil, CEP: 70660-070

margarita@construindojuntos.com.br

Resumo

Este artigo relata uma experiência com a concepção e abordagem de uma disciplina de pós-graduação que a 6 anos vem sendo oferecida e aprimorada no *Mestrado em Gestão do Conhecimento e da Tecnologia da Informação* (MGCTI) da Universidade Católica de Brasília (UCB). Essa disciplina, denominada *Gestão dos Relacionamentos nas Organizações* (GRO), busca introduzir uma visão construtivista para a formação de profissionais de TI com base em redes híbridas de poderosas tecnologias da informação e de impecabilidade na coordenação de compromissos. O artigo consta de uma 4 seções, incluída a introdução. Seção 2 remete a trabalhos e publicações reportados na literatura que buscam evidenciar as competências que a indústria requer dos profissionais de TI e apresenta brevemente a contribuição de Fernando Flores para a compreensão da natureza dos negócios, face às gerações de tecnologias da informação e comunicação que têm sido objeto de desenvolvimento em universidades e centros de pesquisa. Seção 3 relata a experiência, à luz das reflexões anteriores, da adoção da disciplina GRO, com um breve histórico da sua evolução, das principais abordagens que a orientam e do seu conteúdo programático atual. Finalmente a Seção 4 apresenta considerações finais sobre a concepção abordada cuja base se encontra no desenvolvimento de novas competências profissionais e os desdobramentos em termos de pesquisa e estudos que vêm se realizando na UCB.

Palavras chave: atitudes e valores, competências profissionais, perfil profissional de TI, competências conversacionais, organizações, relacionamentos.

Abstract

This article reports an experience with the conception and approach of a graduate discipline which has been offered and improved for the last six years in a Master's Programme in *Knowledge Management and Information Technology* at the Catholic University of Brasília. This discipline, named *Relationships Management in Organizations* (GRO), looks for introducing a constructivist vision to prepare information technology (IT) professionals on the basis of hybrid networks of powerful information technologies mixed up with the impeccability of commitment coordination. The article is composed by an Introduction plus four more sessions. Session 2 points out some related works which look for clarifying industry required skills for IT professionals and a summary of Fernando Flores' contributions to the understanding of the business nature which accompanies the ages of information and communication technologies that have been developed in universities and research centers. Under the previous considerations, Session 3 reports on the experience of the adoption of GRO, along with a short history of its evolution, its main approaches, and its current programmatic contents. Session 4 presents some considerations about the adopted approach whose basis counts on the development of new professional skills.

Keywords: attitudes and values, professional skills, IT professional profile, conversational skills, organizations, relationships.

1. Introdução

Já são muitos os estudos, documentos e outras referências disponíveis sobre a necessidade de aprimoramento do perfil profissional dos egressos de cursos nas áreas da Ciência da Computação e da Tecnologia da Informação, de maneira geral. Tipicamente, o desafio básico posto para nossas Instituições de Ensino Superior (IES) e Centros de Tecnologia (CT) é: que *competências* deveriam ter os profissionais das áreas de TI para fazer frente, com efetividade, às necessidades da Sociedade de maneira geral, e da Indústria, em particular.

A causa principal dessa necessidade de aprimoramento se deve, particularmente, à predominância de um modelo educativo em que não se privilegia o *aprender a aprender*, sendo portanto natural que a implícita defasagem logo se estabeleça. Tal modelo educativo tem, tradicionalmente, se concentrado na transmissão de informação, privilegiando um aspecto da aprendizagem ao qual se costuma referir como "Aprender a Conhecer", e deixa de contemplar aspectos relacionados a outros pilares[1] igualmente (ou até mais) importantes para a aprendizagem de nossos futuros profissionais: o "Aprender a Fazer", o "Aprender a Ser" e o "Aprender a Viver Juntos". E é no arcabouço associado a esses outros pilares, que se encontram os saberes mais consentâneos e sintonizados com as novas competências profissionais requeridas.

É nessa perspectiva que a Universidade Católica de Brasília - UCB – vem, já faz algum tempo, refletindo sobre os projetos pedagógicos dos seus cursos e programas associados à área de TI, de tal forma a promover mudanças significativas em seu modelo educativo de modo a melhor poder acomodar essas tendências. Essas reflexões vêm sendo sustentadas tanto em projetos de pesquisa e dissertações quanto no oferecimento de disciplinas diferenciadas. Com isso, se busca mesclar a aquisição de *conhecimentos* e de *habilidades* técnicas, usualmente associados ao *aprender a conhecer* e ao *aprender a fazer,* respectivamente, com abordagens humanistas (ou não-técnicas), a partir das quais se busca desenvolver todo um conjunto de *atitudes* e *valores* que constituem os tão necessários *aprender a ser* e o *aprender a viver juntos*.

A chamada do XII CIESC, reacendeu a idéia de compartilhar a experiência que vimos desenvolvendo na UCB em fóruns mais abrangentes e consolidados. Vimos então que o relato da nossa experiência atenderia aos temas "*Tópicos o asignaturas específicas*", bem como "*Enseñanza de los aspectos sociales y éticos*". Por outro lado, cumpre informar que a mesma experiência, vista sob a ótica da *Sociedade da Informação* e orientada ao tema de *Cidades Digitais*, foi reportada no que pode ser considerado uma versão preliminar deste artigo, no *Cybercity 2003*, um seminário internacional ocorrido em São Paulo, em outubro/2003.

O artigo presente tem por objetivo relatar a experiência com a concepção e abordagem de uma disciplina de pós-graduação que, já há 6 anos, vimos oferecendo e aprimorando no *Mestrado em Gestão do Conhecimento e da Tecnologia da Informação* – MGCTI – da UCB. Com essa disciplina, denominada *Gestão dos Relacionamentos nas Organizações* – GRO –, busca-se introduzir uma visão construtivista que, adotada na formação de profissionais, venha a resultar em redes de trabalhadores competentes alicerçadas em ágeis redes híbridas de poderosas tecnologias da informação e de impecabilidade na coordenação de compromissos.

O artigo consta de quatro seções, incluindo esta Introdução. A Seção 2 remete a trabalhos e publicações reportados na literatura que buscam evidenciar as competências que a indústria requer dos profissionais de TI. Nessa mesma seção, se apresenta brevemente a contribuição de Fernando Flores [7] para a compreensão da natureza dos negócios face as gerações de tecnologias da informação e comunicação que têm sido objeto de desenvolvimento em universidades e centros de pesquisa. A Seção 3 relata a experiência, à luz das reflexões anteriores, da adoção da disciplina GRO, com um breve histórico da sua evolução, das principais abordagens que a orientam e do seu conteúdo programático atual. Finalmente, na Seção 4, concluímos com algumas considerações sobre os desdobramentos em termos de pesquisa e estudos que vêm se realizando na UCB.

2. Trabalhos correlatos

2.1. Algumas pesquisas e publicações

Dentre a diversidade de estudos e pesquisas que sustentam a necessidade do aprimoramento do perfil dos nossos profissionais de TI, destacamos, a seguir, algumas contribuições nos âmbitos internacional e na-

[1] Esses pilares são apresentados como "*Os quatro pilares da educação*", Capítulo 4 do Relatório "**Educação: Um Tesouro a Descobrir**", recentemente elaborado para a UNESCO por Comissão Internacional sobre educação para o século XXI. (Referência completa: *Educação: um tesouro a descobrir. – São Paulo : Cortez ; Brasília, DF : MEC : UNESCO, 1998*.)

cional.

No nível internacional, citamos, por exemplo, os trabalhos reportados em "*What Industry Needs from Academy*" [9] e em "*New ICT Curricula for the 21st Century. Designing Tomorrow's Education*"[2]. Sobre este último, trata-se de um consórcio, conhecido por *Career Space*, que reúne grandes companhias do setor de Tecnologias de Informação e Comunicação (ICT Companies) tais como: *BT, Cisco Systems, IBM Europe, Intel, Microsoft Europe, Nokia, Nortel Networks, Philips Semiconductors, Siemens AG, Telefonica S.A. and Thales,* mais a *EICTA*, a *European Information and Communications Technology Industry Association.* O Career Space vem trabalhando em parceria com a Comissão Européia...

> *... to encourage and enable more people to join and benefit from a dynamic and exciting e-Europe and to narrow the current skills gap that threatens Europe's prosperity. ...*
> *The Career Space consortium believes that the way in which engineering and computer studies students are educated should change to meet the needs of the ICT industry in the 21st century. It does not presume to tell the University sector how to design curricula, it offers information and suggestions about the needs of the ICT sector and the ways in which the skills gap might be narrowed.*

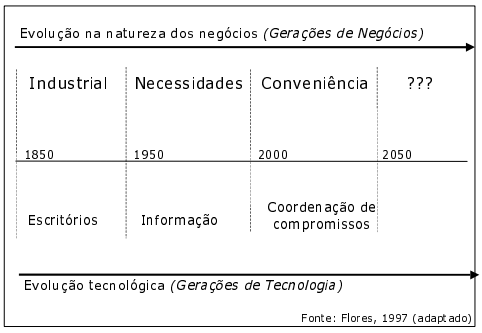
Correspondentemente, no nível nacional, estudos realizados pela *Sociedade Softex – Sociedade para promoção da excelência do software brasileiro* – por exemplo, mostram, com clareza, que esta necessidade de adequação e aprimoramento de perfil profissional também está presente na grande maioria dos profissionais de informática atuantes na indústria brasileira de software.

Por outro lado, estudo de 1998 sobre cenarização, com horizonte no ano 2010, do Ensino Superior (ES) no Brasil, realizado pela Macroplan (www.macroplan.com.br), uma das principais empresas brasileiras especializadas em cenários, preconizava um quadro para o ensino superior com características gerais bastante sintonizadas com o que vem sendo reportado nas referências anteriores. Por outras vias, tipicamente pavimentadas pelo reconhecimento de *incertezas críticas* e *condicionantes de futuro* [10], o estudo revela um conjunto de características tipo *invariantes* (e.g., a "educação como fator crítico para a competitividade das nações, regiões ou empresas), *tendências de peso* (e.g., a "valorização da interdisciplinaridade e do espírito empreendedor como atributos da formação universitária, expansão do ensino a distância") e, por fim, alguns *fatos portadores de futuro* (e.g., a "globalização do ensino e do mercado de trabalho", a necessidade de "certificação profissional periódica" e a "evolução do enfoque de habilidades para das competências"), que, praticamente, corroboram ou fundamentam o juízo corrente de que, de fato, há que se cuidar da manutenção atualizada do perfil profissional do trabalhador de conhecimento do século XXI e, em particular, do profissional do setor de TIC´s.

2.2. Tecnologias e organizações: em que *geração* estamos?

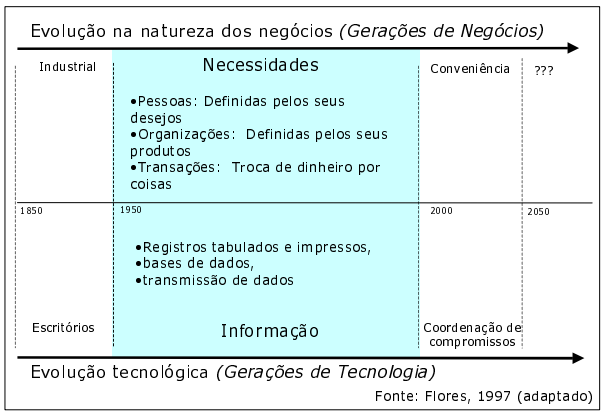
Para localizar o papel das tecnologias nos processos de gestão das organizações, permitam-nos recorrer ao posicionamento de Flores [7], que muito tem inspirado nossas reflexões com a forma convincente e criativa como aborda o fenômeno das *organizações* sob a perspectiva da *comunicação.* Desta vez, estamos recorrendo a uma de suas singulares e sugestivas palestras – *"The Impact of TI on Business"* – proferida em março de 1997, durante as comemorações do 50º aniversário da ACM (*Association for Computing Machinery*), na Califórnia. Nessa palestra, Flores faz uma breve retrospectiva da evolução das tecnologias *versus* seu impacto no mundo dos negócios (ver Figura 1) e nos brinda com uma instigante perspectiva do papel que nos cabe, enquanto *criadores de futuro*, enquanto seres que se constituem também pelo *vir-a-ser* que seremos e enquanto os *history makers* (usando seus próprios termos) que somos enquanto seres humanos e viventes! E tudo isso, numa perspectiva ontológica da *linguagem*, uma perspectiva a partir da qual as organizações, quaisquer que sejam suas origens e domínios de atuação, não passam de *redes de relacionamentos* ou, mais especificamente, de *redes dinâmicas de conversações.* Ver, por exemplo, Flores[6], Winograd & Flores[13] e Spinosa *et al*. [12]. E é nos *relacionamentos* (entre humanos) que se configuram essas redes que nos propomos a investigar o papel da gestão na dinâmica das *organizações*. Com isso, também justificaremos porque nomeamos a disciplina *gestão dos relacionamentos nas organizações*.

Figura 1



[2] *New ICT Curricula for the 21st Century. Designing Tomorrow's Education.* Disponível em http://www.career-space.com/cdguide/serv.htm#top. Acesso em 20 de junho de 2004.

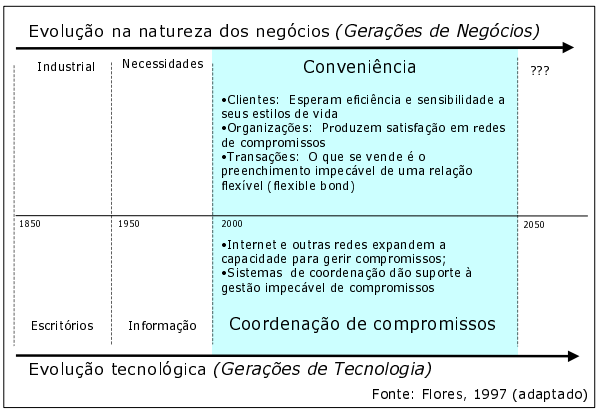
Figura 2



Na perspectiva de Flores (ver Figura 2), o ciclo tecnológico marcado pelas *tecnologias da informação* (TI's) antes da explosão comunicacional provocada pela *Web* (configurando o assim chamado *cyberspace*) teria começado por volta dos anos 50 (com o advento do computador) e estaria terminando por agora, neste final de momento de virada do século e de milênio. Neste ciclo, o uso inicial das TI's – tipicamente, *computadores que tabulam e imprimem registros transacionais, bases de dados que colecionam/coletam informações, dados transmitidos sob forma de mensagens*,... – teria por *finalidade* o preenchimento de *necessidades* de usuários dessas tecnologias. A ***essa geração de tecnologias da informação*** – *antes da Web* – Flores associa uma *geração de negócios* correspondente e que, no caso, dada a natureza da finalidade antes referida, ele chama de ***geração das necessidades***. Nessa geração, o uso das TI's mencionadas estaria orientado a preencher as necessidades das pessoas; na visão das empresas/organizações, as *pessoas* seriam definidas pelos seus *desejos*; as *empresas/organizações* seriam definidas pelos seus *produtos/serviços*; e uma *transação de negócio* (ou *transação organizacional*, no caso de uma organização não empresarial) típica seria a troca de dinheiro por produtos/serviços. Tipicamente, *"necessidade"*, nos termos de Flores, era "um certo tipo de *relação* que o mundo dos negócios tinha com as pessoas. Pessoas tinham desejos, empresas produziam produtos para satisfazer esses desejos e planejavam para aprimorar seus produtos: mais e melhores". Em nosso contexto, estamos estendendo a perspectiva empresarial de Flores, à base de produtos, para uma perspectiva organizacional mais ampla, que, além de empresas, compreenda também órgãos públicos e do terceiro setor e que, assim, desenvolvam e vendam não só produtos mas também serviços.

Ainda na visão de Flores, com a explosão da *Web* (a partir do apagar das luzes do século passado) um mundo novo, um novo tipo de sociedade se descortina, e até mesmo a equação subjacente (e marca de uma época) de que *"mais dados = mais conhecimento = mais poder"* passou a ser questionada. Dados e informações *demais* passaram a ser motivo de preocupações. Novas formas e tipos de *relacionamentos* entre empresas/organizações e clientes surgiram. A cada dia que passa, fica cada vez mais visível que já não basta cuidar das *necessidades* das pessoas; elas continuam querendo tê-las atendidas mas, agora, querem também ***conveniência*** (ver Figura 3). As *transações de negócio/organizacionais*, que antes se estruturavam como não mais que uma troca de dinheiro por produtos/serviços, assumem *novas* características. As pessoas agora querem produtos acompanhados de serviços, de treinamento, de manutenção e, além de tudo, na hora (*just-in-time*) e no lugar (*just-in-place*) em que precisam deles. Nos termos de Flores, esse tipo de *relacionamento* se caracteriza como uma ***promessa*** de atendimento a ***uma ou mais condições de satisfação***.

Figura 3



Para que essa nova modalidade de *transação organizacional* tenha sucesso, é fundamental que o *uso* das agora *Tecnologias de Informação e Comunicação* (TIC´s), principalmente da Internet e outras redes, seja capaz de expandir a capacidade das empresas/organizações para estabelecerem, e cumprirem, *compromissos*. Além disso, é fundamental também que *sistemas de coordenação* dêem suporte à *gestão impecável* desses compromissos. Tais sistemas de coordenação, à base de *TIC´s e gente* (pessoas, seres humanos!) seriam *inovações tecnológicas complementares* [11] que extrapolam o domínio restrito das TIC´s propriamente dito. É através delas que as *redes humanas*, tipicamente *redes de promessas* ou, de maneira mais geral, *redes de relacionamentos*, se estabelecem. Com essas inovações tecnológicas complementares, se torna possível cumprir, atender às expectativas de eficiência e de sensibilidade a *estilos de vida* por parte dos clientes (cidadãos, de maneira geral); com elas se torna possível produzir *satisfação* em *redes de compromissos*, etc. Acreditamos que, com essas tecnologias de suporte à gestão de compromis-

sos, as redes de relacionamentos entre humanos (e máquinas) dentro, fora e inter-organizações tenderão a, ou pelo menos poderão, se tornar cada vez mais harmoniosas e permissivas à criação de uma teia mais ecologicamente correta!

Para mais informações sobre essa perspectiva de *"coordenação de compromissos"* como *base tecnológica* para *transações organizacionais* orientadas às *conveniências* do cliente no domínio empresarial, ver a palestra original já mencionada de Flores [7] e outras obras do mesmo autor [6, 12, 13]. Para os propósitos deste trabalho – nomeadamente a fundamentação de uma visão sistêmica que permeie uma perspectiva de gestão mais comprometida com os compromissos que assumimos quando nos relacionamos – basta que compreendamos o papel e a importância dessas *tecnologias híbridas* (uma mistura de *humanos e máquinas*) *complementares* que vão muito além das TIC´s hoje disponíveis. Investiguemos, portanto, um pouco mais do lado humano dessas tais tecnologias.

3. A Disciplina *GRO*

3.1. Como surge a disciplina

A inquietude pela criação de uma disciplina em que se reflete sobre o fenômeno humano numa área de tecnologias e máquinas surge como conseqüência de inquietações antigas dos autores no âmbito dos *valores* e das *atitudes* com que profissionais, no exercício de suas profissões, em particular, e as pessoas na vida, de modo geral, fazem o que fazem. Nesse contexto, pergunta que nos temos colocado é: ***para que*** **tudo isso?**

Sim, *para que* tudo isso? O que estamos construindo? Com o que estamos comprometidos? com o passado que herdamos ou com o futuro que deixaremos para os nossos filhos e os que virão depois de nós? Ou com os dois?... E, centrando-nos um pouco mais na temática espelhada no artigo, perguntamos: afinal... **gestão, *para que*?** Que papel teria, ou deveriam ter, as tecnologias (não só da informação) na gestão das organizações?

Respondendo a esses questionamentos, a disciplina, se propõe misturar aspectos *humanos*[3] e *tecnológicos*[4]. Desde o começo a disciplina parecia "despertar" algo nos alunos e nos que dela, de alguma forma, se aproximavam. E esse algo, não restava dúvida, estava umbilicalmente ligado à vertente humanística que se explorava no desenvolvimento da disciplina. O que significava (significa) *ser humano* em meio a tanta tecnologia? Que importância tem, ou o que significa, para as organizações, aquela figurinha normalmente encontrada (e esquecida?) entre o teclado de um computador e o costado ou espaldar de uma cadeira, na maioria das vezes, desconfortável?

Quatro ou cinco versões depois e após várias idas e vindas no desenho e redesenho da disciplina, chegamos à conclusão de que, fosse qual fosse seu título, o que desenvolvíamos mesmo era uma *intensa reflexão sobre o que significa ser humano e, particularmente, sobre o que significa ser humano nas organizações.* Daí, sair da viciada denominação de "recursos" (em GRI), dar uma parada na também limitada denominação de "capital" (em GHT) e partir para uma noção mais sistêmica e holística de "relacionamentos" nas organizações foi um pulo! Hoje, a disciplina chama-se *Gestão de Relacionamentos nas Organizações* – GRO. Contar um pouco mais dessa história, compartilhar a riqueza dessa experiência, o desenvolvimento e o estágio atual em que nos encontramos, e para onde estamos indo nessa reflexão, foi o que nos motivou a trazer este relato para o XII CIESC, com a expectativa de que suas sementes possam vir a germinar em searas ibero-americanas.

3.2. Propósito da disciplina

Os objetivos, geral e específicos, que vimos perseguindo com a oferta dessa disciplina na Universidade Católica de Brasília e, mais recentemente, nos mais variados eventos (e.g., seminários, *workshops*, cursos presenciais, cursos a distância, cursos mistos, consultorias) de que temos participado (ora como palestrantes, ora como facilitadores, ora como tutores, ora como consultores, etc.). Em todos os casos, o *objetivo geral* perseguido é sempre:

> *Ajudar a desenvolver nas pessoas a capacidade de se tornarem* observadores diferentes*, mais eficazes e eficientes no trato dos fenômenos organizacionais e, em particular, dos fenômenos associados à*

[3] Inicialmente sob o rótulo de *recursos*, quando a disciplina ainda se chamava *Gestão de Recursos Humanos em Organizações Informatizadas* – GRI – e, posteriormente, como *capital*, quando a disciplina, em sua segunda versão, passou a se chamar *Gestão de Capital Humano e Tecnologia da Informação* – GHT.
[4] Inicialmente sob o rótulo de *informáticos,* em GRI, e, posteriormente, como *tecnologias de informação*, em GHT.

gestão dos relacionamentos nas organizações.

Já em termos de *objetivos específicos*, o que temos buscado sempre é:

- *Desvendar as* competências conversacionais *embutidas na natureza dos processos de* gestão dos relacionamentos nas organizações *e desenvolver a capacidade necessária para realizá-las de modo mais efetivo e, assim, melhorar a prática gerencial do participante.*
- *Desenvolver no participante a capacidade de utilização de algumas* ferramentas básicas *necessárias à prática da gestão dos relacionamentos em suas organizações, ressaltando possibilidades de aplicação imediata destas ferramentas em atividades que o mesmo desenvolva em seu próprio ambiente de trabalho.*

Ao final da disciplina, seus participantes, estarão:

- aptos a compreender que sua *visão de mundo*, por diferenciada que seja, não passa de sua forma de ver o mundo, *atitude* primordial para o exercício de uma ética de *convivência social* centrada no *respeito pelo outro*,
- capazes de *aprender a aprender* e, assim, buscar a *impecabilidade* como padrão básico para sua forma de operar e se conduzir na Vida e, por fim,
- capazes de gerar *resultados*, tanto em suas organizações quanto em suas vidas pessoais, que agreguem valor à qualidade de vida na Sociedade da qual participam.

3.3. Conteúdo Programático

A disciplina é oferecida uma vez por ano no MGCTI da UCB e vale 4 créditos, isto é, tem uma carga horária de 60 horas. Atualmente, os encontros são em número de nove e estão distribuídos sob a forma de 3 *workshops* (de 6 horas de duração, cada), 6 aulas presenciais (de 3 horas de duração, cada) e o restante, gasto com tarefas, reuniões de *grupos de estudo* e reuniões com *equipes de trabalho* (estas constituídas por outras pessoas membros das próprias organizações dos alunos, uma estratégia destinada a fazê-los aprender fazendo, "*hands on*"). Além disso, fóruns virtuais de discussão têm sido progressivamente acrescentados à dinâmica da disciplina, o que tem elevado significativamente sua efetividade. A pauta para o conteúdo programático da disciplina é a seguinte:

UNIDADE 0 – Caracterizando o *momento inicial*

- Quem-é-quem; Levantamento de expectativas: em relação à disciplina e sobre esta em relação ao Curso como um todo.
- Apresentação e discussão do Plano da Disciplina.
- Dando início a uma experiência de aprendizagem

UNIDADE 1 – Organizações, Pessoas e Relacionamentos

- Referenciais teóricos
- O que significa ser humano? O modelo do observador e da ação humana.
- Aprendizagem. Aprendizagem de primeira e segunda ordens. Gestão de Relacionamentos.

UNIDADE 2 - Organizações

- Organizações como redes de relacionamentos.
- Pessoas em relacionamentos peculiares
- O que fazemos quando estamos trabalhando?
- Redes dinâmicas de conversações. A importância das conversações na vida das pessoas e das organizações.
- A Ontologia da Linguagem. Linguagem e ação.
- Avaliação parcial da experiência em curso.

UNIDADE 3 – Pessoas

- Seres humanos como seres lingüísticos.
- Domínios constitutivos do ser humano: Linguagem, Emocionalidade e Corporalidade.
- O Modelo do Observador e da Ação Humana.
- Avaliação parcial da experiência em curso.

UNIDADE 4 - Relacionamentos

- Uma "mirada" conversacional sobre a função gerencial. O que realmente faz um gerente.
- Gerência e conversação.
- Conversações: um entrelaçamento de linguagem, emocionalidade e corporalidade.

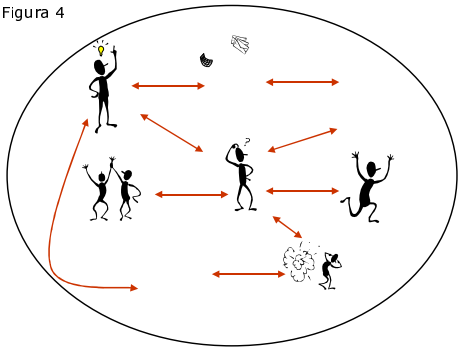
- Tipologias de conversações. Componentes de uma conversação. Desenho de conversações.
- Avaliação parcial da experiência em curso.

UNIDADE 5 – Síntese e Avaliação (Caracterizando o *momento final*)

- Arremate e síntese do que foi desenvolvido nas unidades anteriores.
- Tratamento de eventuais tópicos que possam ter surgido durante o desenvolvimento das Unidades anteriores (e.g., a questão da criatividade na gestão do capital humano). Encaminhamentos.
- Avaliação da experiência de aprendizagem ao final da disciplina.

3.4. Abordagem

Figura 4

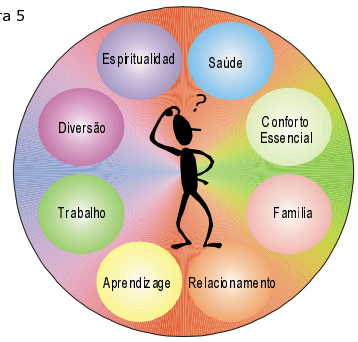


Considere uma organização qualquer, seja ela um órgão público, uma empresa privada, uma família, etc. O que há de mais básico no sentido de melhor caracterizar essa organização? A maioria das pessoas (principalmente hoje em dia quando está na moda falar de gestão de pessoas, etc.), ao se defrontar com essa pergunta, não hesitaria em responder: "claro que são as pessoas! Sem pessoas não há organização!". Entretanto, se pensarmos um pouco mais, veremos que não são pessoas que melhor caracterizam uma organização; o que melhor caracteriza uma organização são os *tipos de relações* que as pessoas (e, às vezes, também máquinas) mantêm entre si. Assim, por exemplo, um mesmo grupo de pessoas (ver figura 4) pode ora relacionar-se como uma família (pai, mãe, filhos, sobrinhas, etc.), ora como uma banda (de música), ora como uma empresa, etc., de modo que o que vai diferenciar a organização da qual estamos falando, em primeiro lugar, são os tipos de relações (setas na figura) mantidos entre as pessoas e, em segundo lugar, as pessoas que fazem parte da mesma. Por outro lado, é bom que fique claro que quando dizemos "primeiro" ou "segundo lugar", não estamos falando de ordem de *importância*; estamos simplesmente querendo denotar o que, basicamente, diferencia uma organização de outra: os *relacionamentos* mantidos pelas pessoas pertencentes a uma organização, dentro desta e a partir desta, com o sistema social com o qual a organização interage.

3.4.1. Pessoas e Organizações: onde estamos?

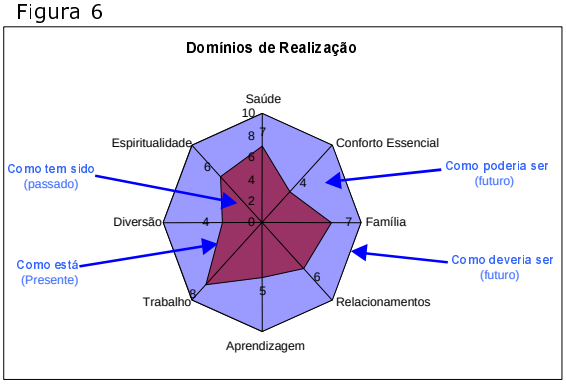
Assim é que, se quisermos compreender a dinâmica de uma organização com o intuito de nela intervirmos visando aumentar ou melhorar sua efetividade, deveremos tanto observar e "mexer" com as *pessoas* quanto com os *relacionamentos* que elas mantêm entre si (seja dentro da organização, seja no âmbito do(s) sistema(s) do(s) qual(ais) a organização faz parte). Nesta subseção, exploraremos a primeira dessas perspectivas, a que tem a ver com as pessoas.

Figura 5



Considere então uma *pessoa típica* dentro da organização, por exemplo, a que está no centro da figura ao lado (ver figura 5), aparentemente, com um bom "problema" na mão. Lembre que essa é uma pessoa típica e que, portanto, a análise que faremos a seguir vale para qualquer outra, participante (ou não) de nossa organização. Essa pessoa, em sua passagem pela vida em nosso pequeno planeta, tipicamente viverá em função de resultados localizados em torno do que chamaremos de *domínios básicos de realização humana: saúde, conforto essencial (moradia), família, relacionamentos, aprendizagem, trabalho, diversão e espiritualidade.* Trata-se de um conjunto arbitrariamente escolhido, com a pretensão de ter características universais (ie., válidas em qualquer região do planeta) e gerais (ie., válidas para qualquer ser humano, independentemente de credos, etnias ou classes sociais). Haveria alguém no mundo que, dada sua natureza (humana) intrínseca, não busque realização em algum desses domínios? Ainda que o tema enseje controvérsias, permitamo-nos seguir com nossa narrativa e consideremos a tal *pessoa típica* nalgum momento (também arbitrário) de sua existência. O que teria ela conquistado, ou construído, ao longo do seu viver, até aquele momento?

Figura 6



Para responder a essa pergunta, usemos uma escala de zero a dez (0 a 10) para demarcar suas conquistas ou seus resultados até então atingidos. Para fins de ilustração, notas arbitrárias são escolhidas e plotadas num diagrama típico como o do desenho na Figura 6. O importante agora seria perguntar: que significado teriam as quatro regiões demarcadas na figura? E aqui pedimos emprestada a interessante taxonomia usada por Hock [8] para moldar suas quatro maneiras de ver as coisas: *como têm sido, como estão, como poderiam ser* e *como deveriam ser*. Esta taxonomia é perfeita para os nossos propósitos nesta exposição: as quatro áreas ilustradas na figura podem ser usadas para demarcar exatamente o mapa existencial daquela *pessoa típica* que escolhemos para falar do fenômeno de ser humano. Conforme indicado, em qualquer momento de sua vida, essa pessoa terá um retrato configurado a partir de como as coisas têm sido, como as coisas estão, como as coisas deveriam ser e como as coisas poderiam ser. E esta classificação, em que pese a simplicidade com que aqui é tratada para falar de fenômenos tão complexos (ser humano, viver) é impressionantemente clara e adequada para descortinar o mundo de possibilidades que visualizamos para construir o tal mundo mais "vivível" a que nos referimos anteriormente. É claro que sempre vão existir pessoas que preferirão mover-se na vida centrando-se no "como as coisas têm sido" (como se estivessem andando de costas); já outras preferirão mover-se única e exclusivamente centradas no "como as coisas estão" (vivendo "aqui e agora"), sem darem muita atenção para o que passou, nem para o que poderá vir; da mesma forma, haverá pessoas que tipicamente viverão em função de "como as coisas deveriam ser" (ora sonhando com um futuro que nunca chega, ora reclamando porque as coisas não são como deveriam!); e, por fim, haverá pessoas que, de olho no *passado*, mas sem descuidar do *futuro*, viverão o *presente* movimentando-se, "rodopiando", como se estivessem dançando (a dança da Vida!), em busca e/ou criando possibilidades; são as pessoas identificadas ou guiadas pelo "como as coisas poderiam ser", segundo a taxonomia em referência.

No que segue, passamos a analisar a dinâmica organizacional a partir da segunda dimensão, anteriormente considerada: *os relacionamentos*.

3.4.2. O modelo do observador e a ação humana

Cremos que é chegada a hora de "darmos sentido" ao que estamos dizendo quando nos referimos à *gestão de compromissos* como *tecnologia-chave* para o estabelecimento de *redes de relacionamentos* harmoniosas e efetivas intra e inter-organizações. Afinal de contas, quando nos relacionamos, o que acontece, *como* o fazemos? Este será nosso objeto de investigação nesta seção.

Se lhes perguntássemos ***"para que lhes pagam?"***, pedindo-lhes para responderem assinalando as ***ações*** básicas que executam quando fazem o que fazem no exercício de suas práticas profissionais (e.g., coordenando projetos, vendendo produtos/serviços, ensinando, liderando, gerenciando, etc.), o que responderiam? Refletindo sobre isso, logo vocês vão se dar conta de que, enquanto trabalhadores do conhecimento (os chamados *knowledge workers*), nós todos somos pagos para *conversar*! Isso mesmo: *conversar*. Ganhamos o pão-de-cada-dia conversando. Vivemos *conversando*. Somos *seres conversacionais*! É através de conversas que nos relacionamos. Conversas que não se limitam apenas em *falar*, nem tampouco só em *ouvir* (uma ação tipicamente biológica). Conversas envolvem *escutar*, uma ação tipicamente lingüística que está sujeita a interpretações e significados; conversas *envolvem emoções, estados de ânimo, gestos e posturas corporais, movimentos, corporalidade!*

Uma outra perspectiva também interessante para salientar a importância das conversas em nossa vida pessoal e no dia-a-dia das organizações seria imaginarmos, por um momento, um dia típico de nossas vidas quando estamos trabalhando. Por favor, imagine-se e observe-se nesse âmbito!

Você notou que a maior parte (99%??!) de nosso tempo no trabalho (na verdade, o mesmo ocorreria noutras organizações de nossas vidas, sejam elas nossa família, nossa escola/universidade, órgãos públicos, lojas comerciais, escritórios que freqüentamos, etc.), é gasta *conversando*? Notou também que nessas *conversas*, sejam elas íntimas ou profissionais, verbais/presenciais ou através de memorandos, cartas, telefonemas, faxes, mensagens eletrônicas, etc., seja qual for o meio de comunicação, estamos sempre fazendo (ou respondendo a) *pedidos*, prometendo ou dando cumprimento (ou não) a *promessas*, fazendo *declarações, afirmações*, emitindo opiniões (ou *juízos*, às vezes nem sempre bem fundamentados), ou

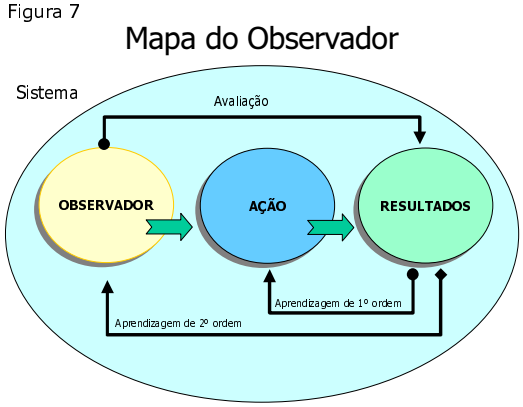
ainda, simplesmente *observando*, lendo, analisando, avaliando, "sentindo" ou, de maneira genérica, "*escutando*" o que se passa ao redor? E mais, em que pesem os títulos, superiores ou não (e por avançados que sejam), percebeu também que em geral não fomos treinados com *habilidades específicas* para manter, de forma efetiva, essas *conversações*? E num último esforço, percebeu também que, além disso e por isso mesmo, quase todos os conflitos que enfrentamos em nosso dia-a-dia – quando estamos nos *relacionando* (!) – se devem a desconfianças, ressentimentos, promessas mal feitas, pedidos mal formulados, juízos equivocados, etc., e que tudo isso, por sua vez, se deve à nossa falta de *competências* para conversar?!!

Quando dizemos que ganhamos a vida conversando, e que conversando nos relacionamos, não temos aí uma tarefa simples. Ao contrário, para conversarmos com efetividade e produzirmos os *resultados* que almejamos, temos diante de nós uma tarefa complexa e delicada. Podemos até querer simplificar (e o fazemos, de vez em quando) mas jamais poderemos ser simplistas (ou simplórios) quando estamos conversando. Disso depende a efetividade de nossos relacionamentos; disso depende a efetividade de nosso viver, pura e simplesmente! Segundo Echeverría [4], na sociedade da informação, ou na sociedade do conhecimento (mais propriamente), o trabalhador do conhecimento ***não*** pode prescindir de ter ***competências conversacionais***; da mesma maneira que na era industrial, em organizações tipicamente *tailoristas*, o trabalhador, para ser considerado produtivo e garantir seu emprego, não podia deixar de ter destreza física, traduzida sob a forma de agilidade, em tempos e movimentos! No que segue, fazemos uma reflexão sobre a forma de *atuar* – uma composição do *pensar*, do *sentir* e do *agir* – do ser humano, através da dinâmica *conversacional* e sobre como esta forma é decisiva na *produção* e na *qualidade* dos *resultados* na vida das pessoas (e das organizações), segundo seus domínios de realização.

Primeiro, é importante compreender que, quando nos referimos a *competências conversacionais,* não estamos nos referindo apenas a ter *conhecimentos* específicos sobre os assuntos dominantes nas conversas, nem tampouco a ter *habilidades* para *falar com clareza* ou para ouvir com atenção e perspicácia; estamos, principalmente, falando de *atitudes* e de *valores* que nos acompanham no uso desses conhecimentos e no exercício dessas habilidades. É a essa combinação de *conhecimentos*, *habilidades* e *atitudes* que Brandão et. al. [2] chamam de *competências*, noção à qual acrescentamos a dimensão *valores*[5] e que seguimos, em nosso caso, instanciando-a no âmbito das conversas nas organizações.

Segundo, ao considerarmos o *atuar* humano como uma composição do pensar, do sentir e do agir – na verdade uma instanciação dos três domínios constitutivos do Ser Humano, a *linguagem*, a *emocionalidade* e a *corporalidade*, segundo a *Ontologia da Linguagem* definida por Echeverría [5] – estamos, em realidade, nos referindo ao seguinte esquema para o atuar humano (ver Figura 7):

Figura 7



Neste esquema – a que nos referiremos como o *mapa do observador* –, entende-se por observador a forma como damos sentido ao que percebemos e distinguimos, por sua vez decorrente do que pensamos e do que sentimos. Ainda segundo o diagrama, de acordo com a nossa capacidade de observar o mundo ou, correspondentemente, de acordo com o que nele percebemos e distinguimos, seremos capazes de agir desta ou daquela forma, o que nos permitirá atingir estes ou aqueles resultados. Ao conjunto Observador/Ação chamamos de ***Pessoa*** Echeverría [5].

Esse esquema, originalmente apresentado por Argyris [1] e posteriormente desenvolvido e adaptado por Echeverría e Pizarro no texto "*El Observador y la Acción Humana*" Echeverría e Pizarro [3], indica que os ***resultados*** atingidos por uma ***pessoa*** (fruto da composição *observador/ação*) num domínio de realização qualquer de sua existência (por exemplo, o domínio do trabalho) dependem, de um lado, de como a *pessoa observa* e *age* (ou, correspondentemente, de como se *comporta*) ao estabelecer suas *interações*, ou seus *relacionamentos*, com o resto do(s) sistema(s) do(s) qual (quais) participa e, de outro lado, de como ela *aprende* na Vida.

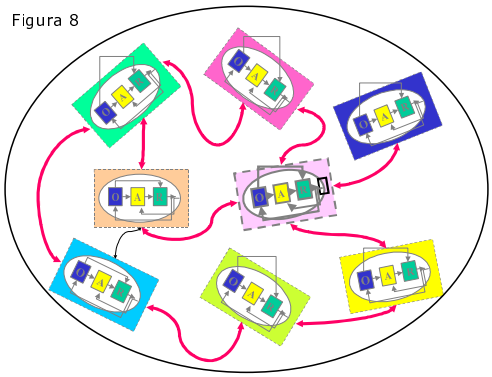
De acordo com o diagrama, quando os resultados são satisfatórios, em geral a pessoa (satisfeita) é levada a validar suas ações e, por conseqüência, sua forma de observar o mundo. Por outro lado, mediante resultados não satisfatórios, negativos (às vezes interpretados como "erros"), o esquema oferece a possibilidade da pessoa *aprender*, ora redesenhando suas *ações* (ao que às vezes se refere como *mudanças táticas*),

[5] Notas de aula de um dos autores. (Trabalho em preparação).

ora redesenhando o *observador* que é ou, correspondentemente, a *forma com que dá sentido às coisas* (ao que às vezes se refere como *mudanças estratégicas*).

Neste ponto, poderíamos nos perguntar: e os *relacionamentos* entre a *pessoa* (observador / ação) e o resto do sistema do qual esta participa, *como* acontecem? Para responder a essa pergunta, visualizemos nosso já conhecido esquema representativo de uma organização qualquer (ie., pessoas em relacionamentos típicos e peculiares; ver Figura 4) e substituamos os "homenzinhos" pelas figuras estruturais equivalentes a partir do mapa do observador, acima introduzido (ver Figura 8).

Figura 8



Este diagrama nos permite visualizar não só o mapa de um observador, mas o mapa de um observador em interações com outros observadores, estruturalmente similares. Interações essas, por sua vez, que se dão a partir de conversas, conversas entre pessoas, ora alegres, ora preocupadas, ora com problemas, ora comemorando, ora trabalhando juntas, etc. como bem mostra as ilustrações (dos "homenzinhos") no diagrama original.

De posse desse modelo, temos agora um mapa a partir do qual podemos raciocinar, compreender e, se necessário, intervir na dinâmica das organizações para, eventualmente, transformá-las em organizações mais ágeis e efetivas. De um lado, podemos nos utilizar do mapa do observador para analisar e compreender como e porque o observador observa o que observa, o que nos levaria a identificar suas *distinções*, seus *juízos*, suas *narrativas*, suas *inquietudes*, suas *atitudes* e *valores*, suas *emocionalidades*, sua *corporalidade*, etc. De outro lado, já no nível das ações, podemos agora nos concentrar nos tipos e formas de *conversações* (ie. nos relacionamentos) que as pessoas mantêm entre si. Com essa combinação, dispomos aí de um instrumental poderoso – normalmente utilizado por *coaches* organizacionais – para, quando necessário, intervirmos na dinâmica operacional das organizações. É com essa noção construtivista, idealizada a partir de uma nova ontologia da linguagem que, pelo lado humano das redes conversacionais em que se constituem as organizações, alimentamos aquela esperança mencionada no início desta apresentação: a de podermos contribuir para a construção de um mundo melhor! A compreensão detalhada do funcionamento dessas redes humanas, alicerçadas por redes tecnológicas avançadas que alavanquem, agilizem e promovam a necessária impecabilidade na *gestão de compromissos* entre humanos (e máquinas) nas organizações, certamente nos facultará a possibilidade dessa construção.

4. Considerações Finais

Em termos práticos, a disciplina cuja história aqui relatamos se propõe e realiza uma nova concepção de organização baseada no desenvolvimento de algumas competências de *trabalho em equipe*, em *competências conversacionais* e em *competências de aprendizagem* (tais como *aprender a conhecer*, a *ser*, a *fazer*, a *conviver*, e a *aprender*), com o objetivo de promover o desenvolvimento de pessoas, de organizações e de seus relacionamentos, com qualidade e satisfação, segundo uma visão integral e holística do ser humano e das organizações.

Desdobramentos dessas abordagens vêm se desenvolvendo também em pesquisas baseadas na linha da gestão do conhecimento sob uma perspectiva da Ontologia da Linguagem, mais particularmente, investigando o papel das competências conversacionais no planejamento e gestão (P&G) das organizações de maneira geral. Em termos mais específicos, e sempre com alunos do MGCTI através de dissertações do mestrado, o trabalho tem sido conduzido no sentido de estudar, compreender, (re-) desenhar e (quando possível) intervir em atividades particulares de P&G nas organizações, a saber: *aquisição do conhecimento, aprendizagem e trabalho em equipe, gestão de relacionamentos, gestão de competências, planejamento estratégico de TI, padrões de linguagem em comunidades virtuais de aprendizagem, avaliação de desempenho/juízos e aprendizagem organizacional/modelos educativos.*

A exemplo da experiência aqui relatada sobre a disciplina GRO, cumpre ressaltar experiência similar que temos levado a efeito com outra disciplina, sobre *Empreendedorismo em Informática*, num curso de Bacharelado em Ciência da Computação, sob o mesmo arcabouço teórico-metodológico. De ambas as experiências, a julgar pelas sistemáticas avaliações a que temos nos submetidos (disciplina e facilitador(es)) a cada vez que a disciplina é ofertada, temos recebido profícuos e incentivadores *feedbacks*.

Esses *feedbacks* têm contribuído para aumentar a crença de que é possível ousar construir um mundo melhor, a partir de uma interação sadia e sustentável, entre humanos e máquinas. Essa construção, fundamentada na idéia de *trabalho cooperativo*, se estabelece a partir de *redes conversacionais* que envolvem pessoas, organizações e tecnologias. Nessas redes, busca-se o estabelecimento de *relacionamentos* que sejam efetivos e satisfatórios a partir dos quais se possa ver manifesta a grandeza humana das pessoas que deles participam, com capacidade criadora, responsabilidade, respeito e amor pelo próximo.

Referências

[1] Argyris, C. Double Loop Learning in Organizations. *Harvard Business Review.* (Sep.-Oct./1977).

[2] Brandão, H.P.; Guimarães, T.A. Gestão de competências e gestão de desempenho: tecnologias distintas ou instrumentos de um mesmo constructo? *Revista de Administração de Empresas*, v.41, n. 1, p. 8-15, jan./mar. 2001.

[3] Echeverría, R.; Pizarro, A.. El Observador y la Acción Humana. Newfield Consulting. México (1987). (Comunicação Pessoal).

[4] Echeverría, R.. A Empresa Emergente. Brasília, Brasil: *Editora Universa*, 2001.

[5] Echeverría, R.. Ontologia del Lenguaje. Santiago, Chile: *Dolmen Ediciones*, 1997 (4ª edição).

[6] Flores, F. Creando Organizaciones para el Futuro. Santiago, Chile: *Dolmen Ediciones*, 1996 (4ª edição).

[7] Flores, F. The Impact of IT on Business. *Conference at the ACM 50th. Anniversary.* California, 1997.

[8] Hock, D. Nascimento da Era Caórdica. São Paulo: *Cultrix e Amana-Key*, 1999.

[9] Jordan, K. What Industry Needs from Academy. CSE Eduacation in 21st Century. CSE at Work: IEEE Computational Science & Engineering, Vol. 3, No. 2, Summer, 1996.

[10] Schwartz, P. The Art of the Long View. Chichester, UK: *Wiley*, 1991 (4th ed.)

[11] Silva, F.Q.B., Lucena Filho, G.J., Kaufman, I., Moura, J.A.B., Marinho, C., Meira, S.R.L. e Aráujo, E.E.R. *Plataforma TIS-BR*. Uma Aposta Estratégica para a Indústria de Tecnologias da Informação e Comunicação no Brasil. Brasil, 2001. Disponível em http://www.softex.br/. Seção Documentos. Acesso em 18 de junho de 2004.

[12] Spinosa, C., Flores, F.; Dreyfus, H. Disclosing New Worlds. *MIT Press*, 1997.

[13] Winograd, T.; Flores, F. Understanding Computers and Cognition. New York: *Addison Wesley*, 1987.

Implementación de una metodología de aprendizaje orientada a la cooperación en un laboratorio de Ingeniería Informática

Alejandro J. Cataldo
Universidad de Atacama, Departamento de Ingeniería Informática y Ciencias de la Computación
Copiapó, Chile, Casilla 240
Email: acataldo@informatica.uda.cl

y

Susana Y. Álvarez
Universidad de Atacama, Departamento de Ingeniería Informática y Ciencias de la Computación
Copiapó, Chile, Casilla 240
Email: salvarezt@labcomp.uda.cl

Abstract

The article that appears next have like objective describe to the experience lived when developing a course of experimental character in which initially a traditional methodology of education for this type of courses was applied. When seeing that the objectives raised in the program were not fulfilled and to notice that the students did not incorporate the knowledge suitably, it was come to make a corrective change in the education methodology so that the cooperative learning between the members of the course was fomented, causing that the roll of the student happened of a passive state to one completely active. Making which this one realized its own process of learning and how to adapt it to the conditions of work in equipment. In this work one will be that simple changes in the form to develop the class can cause significant advances in the profit of direct and cross-sectional objectives such as the promotion of the leadership capacities, the autonomy, the initiative, the construction of learning strategies, among others.

Keywords: Educative methodology, Active Learning, Cooperative Learning.

Resumen

El artículo que se presenta a continuación tiene como objetivo describir la experiencia desarrollada una asignatura de carácter experimental en la que inicialmente se aplicó una metodología tradicional de enseñanza para este tipo de cursos. Al ver que los objetivos planteados en el programa de estudio no se cumplían y notar que los estudiantes no incorporaban adecuadamente los conocimientos, se procedió a realizar un cambio correctivo en la metodología de enseñanza de tal forma que se fomentara el aprendizaje cooperativo entre los miembros del curso, haciendo que el rol del alumno pasara de un estado pasivo a uno completamente activo. Logrando que el alumno se diera cuenta de su propio proceso de aprendizaje y cómo adaptarlo a las condiciones de trabajo en equipo. En este trabajo se mostrará que simples cambios en la forma de desarrollar la clase pueden provocar significativos avances en el logro de objetivos directos y transversales tales como el fomento de las capacidades de liderazgo, la autonomía, la iniciativa, la autoconstrucción de estrategias de aprendizaje, entre otras.

Palabras claves: Metodología de enseñanza, Aprendizaje Cooperativo.

1 Introducción

Desde hace varios años, en la Universidad de Atacama se han estado realizando reformas importantes a las metodologías de enseñanza de las carreras de pedagogía que dicta dicha Casa de Estudios. Esto, principalmente impulsado por el Gobierno de Chile y el proceso de reforma educacional que ha liderado. Los cambios innovadores en las estrategias de enseñanza y aprendizaje desarrolladas se basan fundamentalmente en las metodologías conocidas como ABP o Aprendizaje Basado en Problemas, las que forman parte de las técnicas activas y cooperativas de aprendizaje.

En este contexto es que, dentro de un marco interdisciplinario, estas técnicas, se han aplicado adicionalmente en algunos cursos de la carrera de Ingeniería en Computación e Informática de la misma Casa de Estudios con resultados satisfactorios.

El caso particular que se describirá en este trabajo, se refiere a la aplicación de técnicas de aprendizaje cooperativo desarrolladas en la asignatura de Taller de Conectividad y Sistemas Operativos dictada en el cuarto año de Ingeniería. Como veremos, el uso de estas técnicas didácticas respondió a una corrección metodológica ante la evidencia de resultados pobres en el logro de los objetivos pedagógicos usando métodos tradicionales de enseñanza en una asignatura de tipo experimental. Adicionalmente y al finalizar se presentarán los resultados observados antes y después de realizar la corrección metodológica mencionada las que nos permitirán extraer las conclusiones y recomendaciones que este estudio busca obtener.

2 Marco Conceptual

Cuando se habla de Metodología innovadora, es posible pensar en una forma distinta de enseñar, ahora, podría decirse, que eso depende de qué es lo que vamos a entender por formas de enseñar y qué haría de esto una forma distinta. La verdad es que todos manejamos una idea de cómo enseñar y, es, claro está, la forma en que nosotros fuimos enseñados, el modelo que tendemos a reproducir, en éste, el profesor está en frente de los alumnos y todos los estudiantes atienden, porque es el profesor el que maneja el conocimiento, en esta concepción tradicional de enseñanza, la metodología pasa a ser una serie de supuestos que subyacen a los distintos procedimientos y procesos que ordenan una actividad establecida de forma explícita y repetible con el propósito de lograr algo [1]. En otros casos se alude al estudio de los métodos en sí, es decir, la definición, construcción y validación de los métodos como conjunto de actividades intelectuales que con prescindencia de los contenidos específicos; establece los procedimientos lógicos, formas de razonar, operaciones, procedimientos y reglas que, de una manera ordenada y sistemática, deben seguirse para lograr un fin dado o resultado.

Tomando en cuenta lo expresado en el párrafo anterior es que se ha pretendido adoptar un modelo educativo más actual, el que requiere para su cumplimiento, el uso de nuevas experiencias y por qué no decirlo, el uso de nuevas técnicas didácticas. Las técnicas didácticas son estrategias globales e integrales ya que no son sólo actividades sueltas o sencillas sino que representan un conjunto de actividades ordenadas y articuladas dentro del proceso de enseñanza-aprendizaje de una asignatura [2]. Estas permiten organizar totalmente un curso o bien utilizarse para trabajar en temas específicos dentro de los contenidos del mismo.

La aplicación de las técnicas didácticas que más adelante se presenta, pretende permitir que el alumno ponga en práctica su capacidad de autoaprendizaje asumiendo un rol participativo y colaborativo en el proceso de enseñanza aprendizaje, se comprometa con lo que hace y desarrolle su autonomía [3].

En ella se estimula a los alumnos a participar en el proceso mediante el cual se obtiene el conocimiento. Esto es, se promueve que investiguen por cuenta propia, que analicen la información que han obtenido, estudien cómo un conocimiento se relaciona con otro, sugieran conclusiones, entre otros. Además, la participación en una u otra experiencia con estas técnicas aplicadas permite de una manera vivencial a los alumnos y al profesor hacer énfasis en el conocimiento de la realidad y el compromiso con el entorno, en la medida en que se resuelven ciertas situaciones, se integran problemas, casos o proyectos. Estos ligados al entorno social, cultural y del mundo del trabajo lo que permite una visión más objetiva de la realidad haciendo el aprendizaje de la materia más relevante y profundo [3].

La mayor parte de las técnicas didácticas promueven el desarrollo del aprendizaje colaborativo, estas actividades estimulan la generación de grupos colaborativos entre estudiantes, ya sea de forma presencial o virtual [3].

El docente pone en práctica un nuevo rol: el de facilitar el aprendizaje y hacer que el alumno profundice en los conocimientos. Este cambio en el papel del profesor conlleva una modificación en el papel del alumno: convertirlo en un sujeto activo que construye su conocimiento y adquiere mayor responsabilidad en todos los elementos del proceso.

Todo cambio metodológico trae aparejada una nueva forma de evaluar, cómo se evalúan entonces los aprendizajes desde esta nueva mirada; pues es posible ya no sólo medir el producto, sino también el proceso. Si consideramos que el aprendizaje, en el buen entendido, es un proceso por el cual se crea significación personal a partir de nueva información y de los conocimientos personales que el alumno ya tiene, entonces, podemos evaluar a través de los mismos estudiantes. La participación del alumno en el proceso de evaluación de su aprendizaje se hace de una manera responsable, lo que le permite desarrollar su autonomía, su capacidad de tomar decisiones y de asumir la responsabilidad de las consecuencias de sus actos [3].

Cuando el alumno participa proponiendo, analizando y comparando, tiene la posibilidad de predecir e hipotetizar; además que al trabajar en equipos puede utilizar un nivel de pensamiento crítico que le permite relacionar la información personal con la de los demás.

Por último, nunca está de más el poner de manifiesto que entre los alumnos se encuentra una gran diversidad de individuos que aprenden, con distintos estilos de aprendizaje, rangos atencionales, capacidad de memoria, ritmo de desarrollo e inteligencias [4], considerando esto, es posible también utilizando estas estrategias didácticas, tener la oportunidad de designar roles según las destrezas y los intereses de los alumnos y permitir que ellos tengan la oportunidad para revisar y repensar sobre su forma de adquirir y aplicar el nuevo conocimiento.

3 Descripción de la situación

La asignatura "Taller de conectividad y sistemas operativos" consiste en una cátedra semestral de tipo experimental de cuatro horas semanales que se dicta en el segundo semestre del cuarto año de Ingeniería Civil en Computación e Informática de la Universidad de Atacama. Es una asignatura que corresponde al nuevo plan de Ingeniería Informática por lo que se dictó formalmente, por primera vez, el año 2003.

Al inicio de la asignatura, las actividades fueron estructuradas por medio de una guía práctica entregada previamente (basada en el formato de la Academia de Networking de Cisco System [5]). Esta guía contenía y especificaba explícitamente los objetivos de la experiencia, el tiempo estimado de realización y los pasos que se debían seguir para ejecutar las tareas de la actividad. Previo a cada inicio de sesión, los alumnos debían rendir un control rápido para verificar el estudio de la guía. Los recursos pedagógicos con los que contaban los estudiantes eran computadores y los respectivos equipos de conectividad.

Sin embargo, al transcurrir el semestre, se observó que el aprendizaje de los alumnos no era el esperado, en especial, porque éstos se dedicaban a seguir los pasos descritos en la guía práctica sin internalizar los conocimientos que se buscaba entregar. Por ello, se cambió la metodología de trabajo, orientándose a estrategias de aprendizaje cooperativo. Esto implicó que se reformularan las guías, cambiando su formato, de tal forma de describir un escenario hipotético y plantear un problema a solucionar de acuerdo a este contexto. Se eliminaron las descripciones de pasos a seguir y se pusieron metas a lograr para cada experiencia. En lo que se refiere al ambiente de trabajo, los estudiantes se encontraban con los mismos recursos de siempre pero se adicionó en el laboratorio como recurso de investigación un computador conectado a Internet por grupo.

A continuación, se describe más detalladamente cada situación de aprendizaje. A la primera, estructurada en base a la guía Cisco [5] se le denominará "metodología tradicional", mientras que a la otra se le reconocerá como "metodología orientada a la cooperación".

3.1 Estructura de la asignatura bajo la metodología tradicional

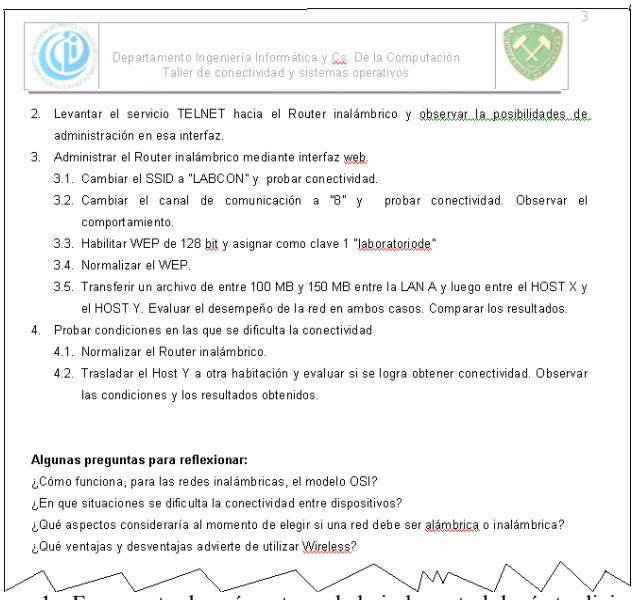
Las prácticas de laboratorio fueron diseñadas siguiendo la metodología tradicional de enseñanza para este tipo de actividades, tal como lo hace el área de entrenamiento de la Academia de Networking de Cisco Systems, perteneciente a Cisco Systems Inc. La guía contempló como parte de su estructura ítems tales como:

- Nombre de la experiencia,
- Duración estimada,
- Aspectos generales (objetivo, conocimientos requeridos, temas acerca de los cuales investigar, herramientas para preparación y recursos necesarios),
- Hoja de trabajo (actividades de la práctica y figuras/gráficos/tablas complementarias aclaratorias),
- Preguntas para reflexionar,
- Una descripción de la estructura general y contenidos del informe final que debía ser entregado como resultado de lo aprendido en el desarrollo de la práctica.

Las prácticas se realizaban siguiendo las instrucciones claramente descritas paso a paso en la sección "Hoja de Trabajo". Para realizar las actividades, a los alumnos se les entregaba la guía una semana antes, con el fin de que estudiaran, reforzaran e investigaran acerca de los conocimientos requeridos para llevar a cabo la experiencia. Al

inicio de cada una de ellas, los estudiantes debían rendir y aprobar un control previo en forma individual, cuyo objetivo era comprobar que los alumnos hubieran estudiado y contaran con los conocimientos requeridos. A continuación, procedían a desarrollar las actividades en forma secuencial paso a paso según eran solicitadas en la guía.

En la figura 1 se muestra una imagen con el fragmento de una guía entregada bajo este esquema de trabajo.



3

Departamento Ingeniería Informática y Cs. De la Computación
Taller de conectividad y sistemas operativos

2. Levantar el servicio TELNET hacia el Router inalámbrico y observar la posibilidades de administración en esa interfaz.
3. Administrar el Router inalámbrico mediante interfaz web.
   3.1. Cambiar el SSID a "LABCON" y probar conectividad.
   3.2. Cambiar el canal de comunicación a "8" y probar conectividad. Observar el comportamiento.
   3.3. Habilitar WEP de 128 bit y asignar como clave 1 "laboratoriode"
   3.4. Normalizar el WEP.
   3.5. Transferir un archivo de entre 100 MB y 150 MB entre la LAN A y luego entre el HOST X y el HOST Y. Evaluar el desempeño de la red en ambos casos. Comparar los resultados.
4. Probar condiciones en las que se dificulta la conectividad
   4.1. Normalizar el Router inalámbrico.
   4.2. Trasladar el Host Y a otra habitación y evaluar si se logra obtener conectividad. Observar las condiciones y los resultados obtenidos.

Algunas preguntas para reflexionar:

¿Cómo funciona, para las redes inalámbricas, el modelo OSI?
¿En que situaciones se dificulta la conectividad entre dispositivos?
¿Qué aspectos consideraría al momento de elegir si una red debe ser alámbrica o inalámbrica?
¿Qué ventajas y desventajas advierte de utilizar Wireless?

Figura 1.- Fragmento de guía entregada bajo la metodología tradicional.

3.2 Estructura de la asignatura bajo la Metodología orientada a la cooperación

Dado que se observó claramente que los estudiantes no estaban incorporando los nuevos conocimientos de acuerdo a la forma esperada y planteada en los objetivos, se procedió, en mitad del semestre a cambiar la metodología usada, por una que fomentara un trabajo cooperativo y que fortaleciera el logro de objetivos por sobre la ejecución de procedimientos secuenciales. Por lo tanto, lo primero que se procedió a hacer fue un cambio en la estructura de la guía tradicional a una que incentivara el logro por actividades en una relación directa entre el planteamiento de desafíos y el cumplimiento de objetivos.

La nueva estructura de la guía para esta nueva metodología incluía ítems como:

- Nombre de la práctica,
- Duración estimada,
- Presentación del problema (una descripción ficticia que justificara la actividad),
- Desafío (planteamientos de las metas esperadas y que serían los indicadores de evaluación),
- Preguntas para reflexionar,
- Una descripción de la estructura general y contenidos del informe final que debía ser entregado como resultado de lo aprendido en el desarrollo de la práctica..

El control para acceder al laboratorio fue eliminado y sólo se confirmó que conocieran de antemano cuáles serían las características del ejercicio que le permitirían cumplir con el objetivo/desafío.

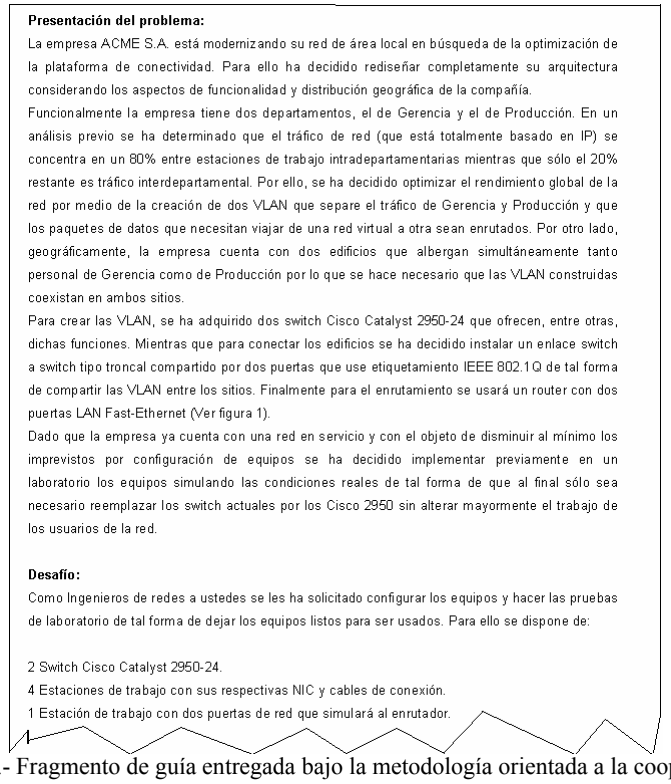
En la figura 2 se muestra una imagen con el fragmento de una guía entregada bajo este esquema de trabajo.

4 Sobre la evaluación

El cambio en la forma de presentar las actividades también implicó una innovación en la forma de evaluar; a continuación se detalla este proceso antes y después de aplicar la metodología orientada a la cooperación.

En un comienzo, bajo la forma tradicional, la calificación era por actividad y quedaba establecida a través de una mezcla proporcional obtenida a partir de los controles previos a cada sesión e informes de actividades al finalizar éstas. Para verificar que lo estudiantes estuvieran preparados en cuanto a conocimiento de la práctica a desarrollar y que manejaran los aspectos teóricos involucrados se estableció un breve control escrito (CtrlPract). Este control servía de filtro de acceso al laboratorio de conectividad y debía ser obligatoriamente resuelto cuya nota no podía ser

inferior a 50%. Si la nota era inferior a 50% implicaba que el alumno no estaba preparado para participar de la práctica y por ende, dada la condición del curso de asistencia obligatoria, la consiguiente reprobación. Tras terminar una experiencia, los alumnos contaban con un plazo que estaba limitado por el inicio de la práctica siguiente para entregar un informe (InfPract). El objetivo del informe era hacer una síntesis de lo aprendido en la experiencia buscando consolidar el conocimiento adquirido. Finalmente la calificación de cada experiencia se obtenía a través de un cálculo simple basado en la ecuación (1).



Presentación del problema:

La empresa ACME S.A. está modernizando su red de área local en búsqueda de la optimización de la plataforma de conectividad. Para ello ha decidido rediseñar completamente su arquitectura considerando los aspectos de funcionalidad y distribución geográfica de la compañía.

Funcionalmente la empresa tiene dos departamentos, el de Gerencia y el de Producción. En un análisis previo se ha determinado que el tráfico de red (que está totalmente basado en IP) se concentra en un 80% entre estaciones de trabajo intradepartamentarias mientras que sólo el 20% restante es tráfico interdepartamental. Por ello, se ha decidido optimizar el rendimiento global de la red por medio de la creación de dos VLAN que separe el tráfico de Gerencia y Producción y que los paquetes de datos que necesitan viajar de una red virtual a otra sean enrutados. Por otro lado, geográficamente, la empresa cuenta con dos edificios que albergan simultáneamente tanto personal de Gerencia como de Producción por lo que se hace necesario que las VLAN construidas coexistan en ambos sitios.

Para crear las VLAN, se ha adquirido dos switch Cisco Catalyst 2950-24 que ofrecen, entre otras, dichas funciones. Mientras que para conectar los edificios se ha decidido instalar un enlace switch a switch tipo troncal compartido por dos puertas que use etiquetamiento IEEE 802.1Q de tal forma de compartir las VLAN entre los sitios. Finalmente para el enrutamiento se usará un router con dos puertas LAN Fast-Ethernet (Ver figura 1).

Dado que la empresa ya cuenta con una red en servicio y con el objeto de disminuir al mínimo los imprevistos por configuración de equipos se ha decidido implementar previamente en un laboratorio los equipos simulando las condiciones reales de tal forma de que al final sólo sea necesario reemplazar los switch actuales por los Cisco 2950 sin alterar mayormente el trabajo de los usuarios de la red.

Desafío:

Como Ingenieros de redes a ustedes se les ha solicitado configurar los equipos y hacer las pruebas de laboratorio de tal forma de dejar los equipos listos para ser usados. Para ello se dispone de:

2 Switch Cisco Catalyst 2950-24.
4 Estaciones de trabajo con sus respectivas NIC y cables de conexión.
1 Estación de trabajo con dos puertas de red que simulará al enrutador.

Figura 2.- Fragmento de guía entregada bajo la metodología orientada a la cooperación.

Cuando se aplicó la metodología orientada a la cooperación, el proceso evaluativo se simplificó notablemente ya que, dado que como se establecían metas, la nota era un reflejo del logro de éstas lo que básicamente se obtenía durante la realización de la actividad. Esto permitió en primer lugar, eliminar la necesidad de los controles previos. Por otra parte, los informes sólo se requerían en términos de dejar un registro de la experiencia por parte de los alumnos. De hecho, el proceso de calificación cambió radicalmente de una forma asíncrona (antes y después de cada experiencia) a una evaluación constante que se obtenía a partir de la observación de las conductas de los alumnos durante la realización de las actividades.

$$NotaFinal = \frac{1}{n}\sum_{i=1}^{n}(Ctrl\,\mathrm{Pr}\,act_i * 0{,}2) + (Inf\,\mathrm{Pr}\,act_i * 0{,}8) \qquad (1)$$

n : Número Total de prácticas realizadas en el semestre.
CtrlPract: Control previo a la práctica i.
InfPract : Informe posterior a la práctica i.

5 Resultados, conductas observadas y discusión

Al comparar las conductas de los estudiantes antes y después de aplicar el ABP fue posible realizar las siguientes observaciones:

1. Aunque bajo la metodología tradicional cada grupo resolvía la guía actuando también en forma cooperativa, especialmente cuando surgían dudas, esta cooperación se centraba básicamente en torno a cómo resolver los problemas de procedimientos que surgían, mientras que bajo la metodología orientada a la cooperación, la colaboración estaba fundada en el logro de la meta final común a todos, es decir, fomentando un trabajo de equipo.
2. Bajo el sistema tradicional los estudiantes sólo se dedicaron a seguir secuencialmente los pasos descritos en la guía, sin que necesariamente se llegara a comprender lo que se estaba haciendo. Bajo la metodología orientada a la cooperación, los estudiantes en forma grupal planificaron previamente los pasos a seguir para

abordar el problema, resolvieron dudas acerca del enunciado, homologaron criterios y diagramaron la posible solución. Esto hizo que abordaran la investigación en forma conjunta, primero utilizando el recurso Internet y aplicándolo a las máquinas.

3. Durante el desarrollo de la actividad misma, todos tuvieron distintas formas de ejecutar sus tareas, algunos se mostraron más analíticos mientras que otros se dedicaron a ejecutar directamente sus ideas sobre los equipos en una especie de método de prueba y error.
4. Un dato interesante es que en una de las actividades obtuvieron su primer logro significativo tras 3 horas de trabajo, lo que provocó una disminución de la ansiedad en la que se encontraban los grupos dándose un breve descanso de algunos minutos y continuando con su tarea. Esta conducta muestra el manejo autónomo del tiempo que ellos mismos se dieron.
5. Al finalizar las actividades se encontraron ante la necesidad de probar si la solución implementada satisfizo los requerimientos (realimentación). Para determinar de qué forma hacerlo, previamente razonaron cómo determinar qué condiciones deben darse, desde el punto de vista teórico, para demostrar que su trabajo fue exitoso y así validar la solución.
6. Cabe destacar que bajo la metodología orientada a la cooperación, algunos grupos mostraron mayor capacidad de trabajo en equipo que otros y mayores grados de tolerancia entre sus integrantes.
7. Finalmente, una vez concluida cada actividad, los estudiantes manifestaron mayor seguridad de los conocimientos aprendidos.

Interesante fue observar cómo de lo que antes era solo seguir instrucciones fue reemplazado por la formulación de estrategias grupales de resolución de problemas (Ej. si sabían donde encontrar la información requerida directamente la localizaban y utilizaban, si no lo sabían se dividían la investigación en equipos siempre de dos estudiantes como mínimo para que cooperaran y discutieran sus hallazgos entre sí), uso de esquemas para buscar alternativas de solución, liderazgo para la organización de las tareas, liderazgo del conocimiento, credibilidad en el aporte de los pares y seguridad ante los conocimientos adquiridos.

Finalmente, a todos los grupos se les preguntó con qué técnica educativa sintieron que habían aprendido mejor y unánimemente manifestaron que bajo la nueva metodología, ya que se basaba en el logro de objetivo/desafío lo que es similar al ambiente laboral. Además que les permitió percatarse de sus propias falencias individuales y trabajar en ellas apoyados por sus compañeros.

6 Conclusiones

En este trabajo se ha presentado la experiencia exitosa de aplicar un método de enseñanza orientado a la cooperación en un laboratorio de una asignatura de Ingeniería Informática. Esto debido a que se pudo observar a tiempo que la propuesta tradicional de cómo se realizan este tipo de asignatura no permitió producir un cambio conductual significativo en los estudiantes (aprendizaje). La propuesta nueva estuvo orientada a fortalecer las técnicas cooperativas y que el alumno fuera el protagonista de su propio aprendizaje (aprendizaje activo) en concordancia con las teorías contructivistas del conocimiento.

Bajo la metodología tradicional, el protagonismo en la clase estaba ubicado en la guía de trabajo, es decir, indirectamente en el profesor. Mientras que bajo la metodología orientada a la cooperación, este rol principal es cedido al estudiante como centro del proceso de aprendizaje y el cumplimiento del objetivo/desafío como la prueba irrefutable del conocimiento adquirido.

El cambio del método tradicional al orientado a la cooperación, trajo consigo que los estudiantes actuaran en forma grupal y realizaran un trabajo colaborativo, integrando sus conocimientos (adquiridos aparentemente en forma aislada), creando sus propias estrategias de resolución de problemas, planificando previamente los pasos a seguir para abordarlos, resolviendo dudas acerca del enunciado, homologando criterios y representando la o las posibles soluciones. Además permitió fortalecer y desarrollar habilidades tales como liderazgo, capacidad de trabajo en equipo, razonamiento crítico, autonomía y autoaprendizaje, todas ellas muy valoradas por el mercado laboral y por los estudios de programas curriculares para ingeniería. Todo esto hizo que los alumnos sintieran y manifestaran mayor seguridad en sus conocimientos sobre la materia.

Hay que decir que el método recibió la aprobación de todos los integrantes del curso en forma unánime lo que lo validó también ante ellos haciendo que apreciaran los cambios de una forma positiva.

Por lo tanto, se puede concluir que al aplicar una metodología como la que se presentó, es posible observar una experiencia exitosa de enseñanza.

Referencias

[1] Ander-Egg, Ezequiel. DICCIONARIO DE PEDAGOGÍA. 1ª edición. Buenos Aires, Argentina. 1997.
[2] Instituto Tecnológico y de Estudios Superiores de Monterrey. EL APRENDIZAJE BASADO EN PROBLEMAS COMO TÉCNICA DIDÁCTICA, http://www.sistema.itesm.mx/va/dide/inf-doc/estrategias. 2000.
[3] Cataldo, Alejandro – Aliaga, Verónica. METODOLOGÍAS INNOVADORAS DE ENSEÑANZA Y CENTRADAS EN EL ALUMNO EN LA CARRERA DE INGENIERÍA INFORMÁTICA EN LA UNIVERSIDAD DE ATACAMA. (Por editar). I Simposio Iberoamericano de Educación, Cibernética e Informática (SIECI 2004). Orlando, EE.UU. 2004.
[4] Aliaga, Verónica. INNOVACIONES EN PEDAGOGÍA UNIVERSITARIA. V Encuentro Nacional de Psicología Cognitiva. Santiago, Chile. 2003.
[5] Lorenz, Jim. Cisco Systems Inc. ACADEMIA DE NETWORKING DE CISCO SYSTEM - PRÁCTICAS DE LABORATORIO (VOLUMEN I). 2ª edición. Pearson Educación. Madrid, España. 2002.

Implementación de una metodología de aprendizaje orientada a la cooperación en un laboratorio de Ingeniería Informática

Alejandro J. Cataldo
Universidad de Atacama, Departamento de Ingeniería Informática y Ciencias de la Computación
Copiapó, Chile, Casilla 240
Email: acataldo@informatica.uda.cl

y

Susana Y. Álvarez
Universidad de Atacama, Departamento de Ingeniería Informática y Ciencias de la Computación
Copiapó, Chile, Casilla 240
Email: salvarezt@labcomp.uda.cl

Abstract

The article that appears next have like objective describe to the experience lived when developing a course of experimental character in which initially a traditional methodology of education for this type of courses was applied. When seeing that the objectives raised in the program were not fulfilled and to notice that the students did not incorporate the knowledge suitably, it was come to make a corrective change in the education methodology so that the cooperative learning between the members of the course was fomented, causing that the roll of the student happened of a passive state to one completely active. Making which this one realized its own process of learning and how to adapt it to the conditions of work in equipment. In this work one will be that simple changes in the form to develop the class can cause significant advances in the profit of direct and cross-sectional objectives such as the promotion of the leadership capacities, the autonomy, the initiative, the construction of learning strategies, among others.

Keywords: Educative methodology, Active Learning, Cooperative Learning.

Resumen

El artículo que se presenta a continuación tiene como objetivo describir la experiencia desarrollada una asignatura de carácter experimental en la que inicialmente se aplicó una metodología tradicional de enseñanza para este tipo de cursos. Al ver que los objetivos planteados en el programa de estudio no se cumplían y notar que los estudiantes no incorporaban adecuadamente los conocimientos, se procedió a realizar un cambio correctivo en la metodología de enseñanza de tal forma que se fomentara el aprendizaje cooperativo entre los miembros del curso, haciendo que el rol del alumno pasara de un estado pasivo a uno completamente activo. Logrando que el alumno se diera cuenta de su propio proceso de aprendizaje y cómo adaptarlo a las condiciones de trabajo en equipo. En este trabajo se mostrará que simples cambios en la forma de desarrollar la clase pueden provocar significativos avances en el logro de objetivos directos y transversales tales como el fomento de las capacidades de liderazgo, la autonomía, la iniciativa, la autoconstrucción de estrategias de aprendizaje, entre otras.

Palabras claves: Metodología de enseñanza, Aprendizaje Cooperativo.

1 Introducción

Desde hace varios años, en la Universidad de Atacama se han estado realizando reformas importantes a las metodologías de enseñanza de las carreras de pedagogía que dicta dicha Casa de Estudios. Esto, principalmente impulsado por el Gobierno de Chile y el proceso de reforma educacional que ha liderado. Los cambios innovadores en las estrategias de enseñanza y aprendizaje desarrolladas se basan fundamentalmente en las metodologías conocidas como ABP o Aprendizaje Basado en Problemas, las que forman parte de las técnicas activas y cooperativas de aprendizaje.

En este contexto es que, dentro de un marco interdisciplinario, estas técnicas, se han aplicado adicionalmente en algunos cursos de la carrera de Ingeniería en Computación e Informática de la misma Casa de Estudios con resultados satisfactorios.

El caso particular que se describirá en este trabajo, se refiere a la aplicación de técnicas de aprendizaje cooperativo desarrolladas en la asignatura de Taller de Conectividad y Sistemas Operativos dictada en el cuarto año de Ingeniería. Como veremos, el uso de estas técnicas didácticas respondió a una corrección metodológica ante la evidencia de resultados pobres en el logro de los objetivos pedagógicos usando métodos tradicionales de enseñanza en una asignatura de tipo experimental. Adicionalmente y al finalizar se presentarán los resultados observados antes y después de realizar la corrección metodológica mencionada las que nos permitirán extraer las conclusiones y recomendaciones que este estudio busca obtener.

2 Marco Conceptual

Cuando se habla de Metodología innovadora, es posible pensar en una forma distinta de enseñar, ahora, podría decirse, que eso depende de qué es lo que vamos a entender por formas de enseñar y qué haría de esto una forma distinta. La verdad es que todos manejamos una idea de cómo enseñar y, es, claro está, la forma en que nosotros fuimos enseñados, el modelo que tendemos a reproducir, en éste, el profesor está en frente de los alumnos y todos los estudiantes atienden, porque es el profesor el que maneja el conocimiento, en esta concepción tradicional de enseñanza, la metodología pasa a ser una serie de supuestos que subyacen a los distintos procedimientos y procesos que ordenan una actividad establecida de forma explícita y repetible con el propósito de lograr algo [1]. En otros casos se alude al estudio de los métodos en sí, es decir, la definición, construcción y validación de los métodos como conjunto de actividades intelectuales que con prescindencia de los contenidos específicos; establece los procedimientos lógicos, formas de razonar, operaciones, procedimientos y reglas que, de una manera ordenada y sistemática, deben seguirse para lograr un fin dado o resultado.

Tomando en cuenta lo expresado en el párrafo anterior es que se ha pretendido adoptar un modelo educativo más actual, el que requiere para su cumplimiento, el uso de nuevas experiencias y por qué no decirlo, el uso de nuevas técnicas didácticas. Las técnicas didácticas son estrategias globales e integrales ya que no son sólo actividades sueltas o sencillas sino que representan un conjunto de actividades ordenadas y articuladas dentro del proceso de enseñanza-aprendizaje de una asignatura [2]. Estas permiten organizar totalmente un curso o bien utilizarse para trabajar en temas específicos dentro de los contenidos del mismo.

La aplicación de las técnicas didácticas que más adelante se presenta, pretende permitir que el alumno ponga en práctica su capacidad de autoaprendizaje asumiendo un rol participativo y colaborativo en el proceso de enseñanza aprendizaje, se comprometa con lo que hace y desarrolle su autonomía [3].

En ella se estimula a los alumnos a participar en el proceso mediante el cual se obtiene el conocimiento. Esto es, se promueve que investiguen por cuenta propia, que analicen la información que han obtenido, estudien cómo un conocimiento se relaciona con otro, sugieran conclusiones, entre otros. Además, la participación en una u otra experiencia con estas técnicas aplicadas permite de una manera vivencial a los alumnos y al profesor hacer énfasis en el conocimiento de la realidad y el compromiso con el entorno, en la medida en que se resuelven ciertas situaciones, se integran problemas, casos o proyectos. Estos ligados al entorno social, cultural y del mundo del trabajo lo que permite una visión más objetiva de la realidad haciendo el aprendizaje de la materia más relevante y profundo [3].

La mayor parte de las técnicas didácticas promueven el desarrollo del aprendizaje colaborativo, estas actividades estimulan la generación de grupos colaborativos entre estudiantes, ya sea de forma presencial o virtual [3].

El docente pone en práctica un nuevo rol: el de facilitar el aprendizaje y hacer que el alumno profundice en los conocimientos. Este cambio en el papel del profesor conlleva una modificación en el papel del alumno: convertirlo en un sujeto activo que construye su conocimiento y adquiere mayor responsabilidad en todos los elementos del proceso.

Todo cambio metodológico trae aparejada una nueva forma de evaluar, cómo se evalúan entonces los aprendizajes desde esta nueva mirada; pues es posible ya no sólo medir el producto, sino también el proceso. Si consideramos que el aprendizaje, en el buen entendido, es un proceso por el cual se crea significación personal a partir de nueva información y de los conocimientos personales que el alumno ya tiene, entonces, podemos evaluar a través de los mismos estudiantes. La participación del alumno en el proceso de evaluación de su aprendizaje se hace de una manera responsable, lo que le permite desarrollar su autonomía, su capacidad de tomar decisiones y de asumir la responsabilidad de las consecuencias de sus actos [3].

Cuando el alumno participa proponiendo, analizando y comparando, tiene la posibilidad de predecir e hipotetizar; además que al trabajar en equipos puede utilizar un nivel de pensamiento crítico que le permite relacionar la información personal con la de los demás.

Por último, nunca está de más el poner de manifiesto que entre los alumnos se encuentra una gran diversidad de individuos que aprenden, con distintos estilos de aprendizaje, rangos atencionales, capacidad de memoria, ritmo de desarrollo e inteligencias [4], considerando esto, es posible también utilizando estas estrategias didácticas, tener la oportunidad de designar roles según las destrezas y los intereses de los alumnos y permitir que ellos tengan la oportunidad para revisar y repensar sobre su forma de adquirir y aplicar el nuevo conocimiento.

3 Descripción de la situación

La asignatura "Taller de conectividad y sistemas operativos" consiste en una cátedra semestral de tipo experimental de cuatro horas semanales que se dicta en el segundo semestre del cuarto año de Ingeniería Civil en Computación e Informática de la Universidad de Atacama. Es una asignatura que corresponde al nuevo plan de Ingeniería Informática por lo que se dictó formalmente, por primera vez, el año 2003.

Al inicio de la asignatura, las actividades fueron estructuradas por medio de una guía práctica entregada previamente (basada en el formato de la Academia de Networking de Cisco System [5]). Esta guía contenía y especificaba explícitamente los objetivos de la experiencia, el tiempo estimado de realización y los pasos que se debían seguir para ejecutar las tareas de la actividad. Previo a cada inicio de sesión, los alumnos debían rendir un control rápido para verificar el estudio de la guía. Los recursos pedagógicos con los que contaban los estudiantes eran computadores y los respectivos equipos de conectividad.

Sin embargo, al transcurrir el semestre, se observó que el aprendizaje de los alumnos no era el esperado, en especial, porque éstos se dedicaban a seguir los pasos descritos en la guía práctica sin internalizar los conocimientos que se buscaba entregar. Por ello, se cambió la metodología de trabajo, orientándose a estrategias de aprendizaje cooperativo. Esto implicó que se reformularan las guías, cambiando su formato, de tal forma de describir un escenario hipotético y plantear un problema a solucionar de acuerdo a este contexto. Se eliminaron las descripciones de pasos a seguir y se pusieron metas a lograr para cada experiencia. En lo que se refiere al ambiente de trabajo, los estudiantes se encontraban con los mismos recursos de siempre pero se adicionó en el laboratorio como recurso de investigación un computador conectado a Internet por grupo.

A continuación, se describe más detalladamente cada situación de aprendizaje. A la primera, estructurada en base a la guía Cisco [5] se le denominará "metodología tradicional", mientras que a la otra se le reconocerá como "metodología orientada a la cooperación".

3.1 Estructura de la asignatura bajo la metodología tradicional

Las prácticas de laboratorio fueron diseñadas siguiendo la metodología tradicional de enseñanza para este tipo de actividades, tal como lo hace el área de entrenamiento de la Academia de Networking de Cisco Systems, perteneciente a Cisco Systems Inc. La guía contempló como parte de su estructura ítems tales como:

- Nombre de la experiencia,
- Duración estimada,
- Aspectos generales (objetivo, conocimientos requeridos, temas acerca de los cuales investigar, herramientas para preparación y recursos necesarios),
- Hoja de trabajo (actividades de la práctica y figuras/gráficos/tablas complementarias aclaratorias),
- Preguntas para reflexionar,
- Una descripción de la estructura general y contenidos del informe final que debía ser entregado como resultado de lo aprendido en el desarrollo de la práctica.

Las prácticas se realizaban siguiendo las instrucciones claramente descritas paso a paso en la sección "Hoja de Trabajo". Para realizar las actividades, a los alumnos se les entregaba la guía una semana antes, con el fin de que estudiaran, reforzaran e investigaran acerca de los conocimientos requeridos para llevar a cabo la experiencia. Al

inicio de cada una de ellas, los estudiantes debían rendir y aprobar un control previo en forma individual, cuyo objetivo era comprobar que los alumnos hubieran estudiado y contaran con los conocimientos requeridos. A continuación, procedían a desarrollar las actividades en forma secuencial paso a paso según eran solicitadas en la guía.

En la figura 1 se muestra una imagen con el fragmento de una guía entregada bajo este esquema de trabajo.

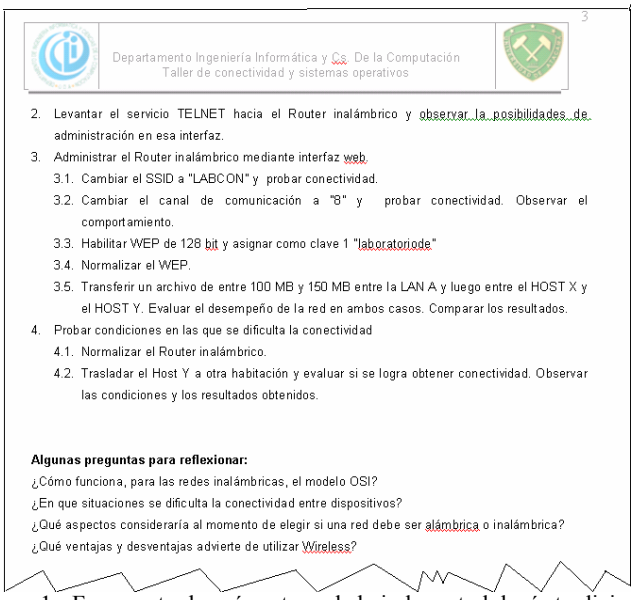


Figura 1.- Fragmento de guía entregada bajo la metodología tradicional.

3.2 Estructura de la asignatura bajo la Metodología orientada a la cooperación

Dado que se observó claramente que los estudiantes no estaban incorporando los nuevos conocimientos de acuerdo a la forma esperada y planteada en los objetivos, se procedió, en mitad del semestre a cambiar la metodología usada, por una que fomentara un trabajo cooperativo y que fortaleciera el logro de objetivos por sobre la ejecución de procedimientos secuenciales. Por lo tanto, lo primero que se procedió a hacer fue un cambio en la estructura de la guía tradicional a una que incentivara el logro por actividades en una relación directa entre el planteamiento de desafíos y el cumplimiento de objetivos.

La nueva estructura de la guía para esta nueva metodología incluía ítems como:

- Nombre de la práctica,
- Duración estimada,
- Presentación del problema (una descripción ficticia que justificara la actividad),
- Desafío (planteamientos de las metas esperadas y que serían los indicadores de evaluación),
- Preguntas para reflexionar,
- Una descripción de la estructura general y contenidos del informe final que debía ser entregado como resultado de lo aprendido en el desarrollo de la práctica..

El control para acceder al laboratorio fue eliminado y sólo se confirmó que conocieran de antemano cuáles serían las características del ejercicio que le permitirían cumplir con el objetivo/desafío.

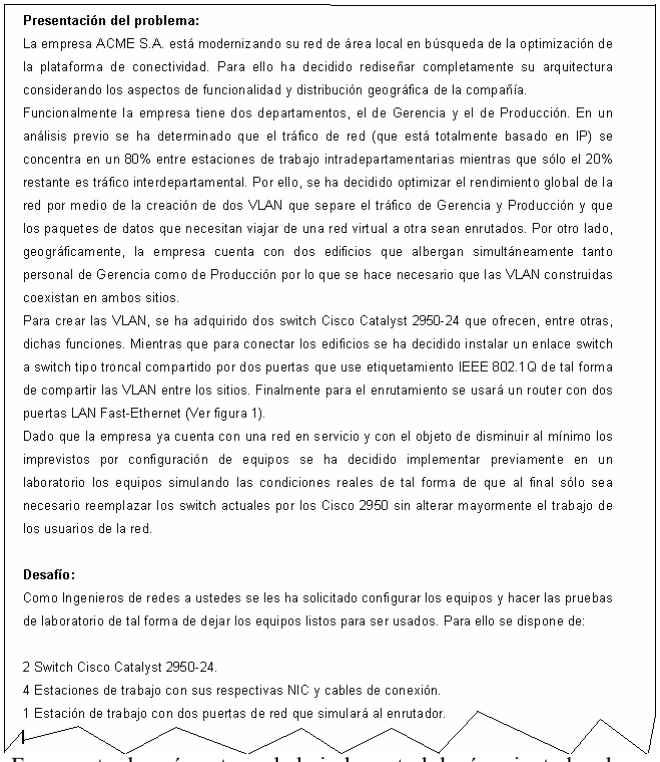
En la figura 2 se muestra una imagen con el fragmento de una guía entregada bajo este esquema de trabajo.

4 Sobre la evaluación

El cambio en la forma de presentar las actividades también implicó una innovación en la forma de evaluar; a continuación se detalla este proceso antes y después de aplicar la metodología orientada a la cooperación.

En un comienzo, bajo la forma tradicional, la calificación era por actividad y quedaba establecida a través de una mezcla proporcional obtenida a partir de los controles previos a cada sesión e informes de actividades al finalizar éstas. Para verificar que lo estudiantes estuvieran preparados en cuanto a conocimiento de la práctica a desarrollar y que manejaran los aspectos teóricos involucrados se estableció un breve control escrito (CtrlPract). Este control servía de filtro de acceso al laboratorio de conectividad y debía ser obligatoriamente resuelto cuya nota no podía ser

inferior a 50%. Si la nota era inferior a 50% implicaba que el alumno no estaba preparado para participar de la práctica y por ende, dada la condición del curso de asistencia obligatoria, la consiguiente reprobación. Tras terminar una experiencia, los alumnos contaban con un plazo que estaba limitado por el inicio de la práctica siguiente para entregar un informe (InfPract). El objetivo del informe era hacer una síntesis de lo aprendido en la experiencia buscando consolidar el conocimiento adquirido. Finalmente la calificación de cada experiencia se obtenía a través de un cálculo simple basado en la ecuación (1).



Presentación del problema:

La empresa ACME S.A. está modernizando su red de área local en búsqueda de la optimización de la plataforma de conectividad. Para ello ha decidido rediseñar completamente su arquitectura considerando los aspectos de funcionalidad y distribución geográfica de la compañía.

Funcionalmente la empresa tiene dos departamentos, el de Gerencia y el de Producción. En un análisis previo se ha determinado que el tráfico de red (que está totalmente basado en IP) se concentra en un 80% entre estaciones de trabajo intradepartamentarias mientras que sólo el 20% restante es tráfico interdepartamental. Por ello, se ha decidido optimizar el rendimiento global de la red por medio de la creación de dos VLAN que separe el tráfico de Gerencia y Producción y que los paquetes de datos que necesitan viajar de una red virtual a otra sean enrutados. Por otro lado, geográficamente, la empresa cuenta con dos edificios que albergan simultáneamente tanto personal de Gerencia como de Producción por lo que se hace necesario que las VLAN construidas coexistan en ambos sitios.

Para crear las VLAN, se ha adquirido dos switch Cisco Catalyst 2950-24 que ofrecen, entre otras, dichas funciones. Mientras que para conectar los edificios se ha decidido instalar un enlace switch a switch tipo troncal compartido por dos puertas que use etiquetamiento IEEE 802.1Q de tal forma de compartir las VLAN entre los sitios. Finalmente para el enrutamiento se usará un router con dos puertas LAN Fast-Ethernet (Ver figura 1).

Dado que la empresa ya cuenta con una red en servicio y con el objeto de disminuir al mínimo los imprevistos por configuración de equipos se ha decidido implementar previamente en un laboratorio los equipos simulando las condiciones reales de tal forma de que al final sólo sea necesario reemplazar los switch actuales por los Cisco 2950 sin alterar mayormente el trabajo de los usuarios de la red.

Desafío:

Como Ingenieros de redes a ustedes se les ha solicitado configurar los equipos y hacer las pruebas de laboratorio de tal forma de dejar los equipos listos para ser usados. Para ello se dispone de:

2 Switch Cisco Catalyst 2950-24.
4 Estaciones de trabajo con sus respectivas NIC y cables de conexión.
1 Estación de trabajo con dos puertas de red que simulará al enrutador.

Figura 2.- Fragmento de guía entregada bajo la metodología orientada a la cooperación.

Cuando se aplicó la metodología orientada a la cooperación, el proceso evaluativo se simplificó notablemente ya que, dado que como se establecían metas, la nota era un reflejo del logro de éstas lo que básicamente se obtenía durante la realización de la actividad. Esto permitió en primer lugar, eliminar la necesidad de los controles previos. Por otra parte, los informes sólo se requerían en términos de dejar un registro de la experiencia por parte de los alumnos. De hecho, el proceso de calificación cambió radicalmente de una forma asíncrona (antes y después de cada experiencia) a una evaluación constante que se obtenía a partir de la observación de las conductas de los alumnos durante la realización de las actividades.

$$NotaFinal = \frac{1}{n}\sum_{i=1}^{n}(Ctrl\,\mathrm{Pr}\,act_i * 0{,}2) + (Inf\,\mathrm{Pr}\,act_i * 0{,}8) \qquad (1)$$

n : Número Total de prácticas realizadas en el semestre.
CtrlPract: Control previo a la práctica i.
InfPract : Informe posterior a la práctica i.

5 Resultados, conductas observadas y discusión

Al comparar las conductas de los estudiantes antes y después de aplicar el ABP fue posible realizar las siguientes observaciones:

1. Aunque bajo la metodología tradicional cada grupo resolvía la guía actuando también en forma cooperativa, especialmente cuando surgían dudas, esta cooperación se centraba básicamente en torno a cómo resolver los problemas de procedimientos que surgían, mientras que bajo la metodología orientada a la cooperación, la colaboración estaba fundada en el logro de la meta final común a todos, es decir, fomentando un trabajo de equipo.
2. Bajo el sistema tradicional los estudiantes sólo se dedicaron a seguir secuencialmente los pasos descritos en la guía, sin que necesariamente se llegara a comprender lo que se estaba haciendo. Bajo la metodología orientada a la cooperación, los estudiantes en forma grupal planificaron previamente los pasos a seguir para

abordar el problema, resolvieron dudas acerca del enunciado, homologaron criterios y diagramaron la posible solución. Esto hizo que abordaran la investigación en forma conjunta, primero utilizando el recurso Internet y aplicándolo a las máquinas.

3. Durante el desarrollo de la actividad misma, todos tuvieron distintas formas de ejecutar sus tareas, algunos se mostraron más analíticos mientras que otros se dedicaron a ejecutar directamente sus ideas sobre los equipos en una especie de método de prueba y error.
4. Un dato interesante es que en una de las actividades obtuvieron su primer logro significativo tras 3 horas de trabajo, lo que provocó una disminución de la ansiedad en la que se encontraban los grupos dándose un breve descanso de algunos minutos y continuando con su tarea. Esta conducta muestra el manejo autónomo del tiempo que ellos mismos se dieron.
5. Al finalizar las actividades se encontraron ante la necesidad de probar si la solución implementada satisfizo los requerimientos (realimentación). Para determinar de qué forma hacerlo, previamente razonaron cómo determinar qué condiciones deben darse, desde el punto de vista teórico, para demostrar que su trabajo fue exitoso y así validar la solución.
6. Cabe destacar que bajo la metodología orientada a la cooperación, algunos grupos mostraron mayor capacidad de trabajo en equipo que otros y mayores grados de tolerancia entre sus integrantes.
7. Finalmente, una vez concluida cada actividad, los estudiantes manifestaron mayor seguridad de los conocimientos aprendidos.

Interesante fue observar cómo de lo que antes era solo seguir instrucciones fue reemplazado por la formulación de estrategias grupales de resolución de problemas (Ej. si sabían donde encontrar la información requerida directamente la localizaban y utilizaban, si no lo sabían se dividían la investigación en equipos siempre de dos estudiantes como mínimo para que cooperaran y discutieran sus hallazgos entre sí), uso de esquemas para buscar alternativas de solución, liderazgo para la organización de las tareas, liderazgo del conocimiento, credibilidad en el aporte de los pares y seguridad ante los conocimientos adquiridos.

Finalmente, a todos los grupos se les preguntó con qué técnica educativa sintieron que habían aprendido mejor y unánimemente manifestaron que bajo la nueva metodología, ya que se basaba en el logro de objetivo/desafío lo que es similar al ambiente laboral. Además que les permitió percatarse de sus propias falencias individuales y trabajar en ellas apoyados por sus compañeros.

6 Conclusiones

En este trabajo se ha presentado la experiencia exitosa de aplicar un método de enseñanza orientado a la cooperación en un laboratorio de una asignatura de Ingeniería Informática. Esto debido a que se pudo observar a tiempo que la propuesta tradicional de cómo se realizan este tipo de asignatura no permitió producir un cambio conductual significativo en los estudiantes (aprendizaje). La propuesta nueva estuvo orientada a fortalecer las técnicas cooperativas y que el alumno fuera el protagonista de su propio aprendizaje (aprendizaje activo) en concordancia con las teorías contructivistas del conocimiento.

Bajo la metodología tradicional, el protagonismo en la clase estaba ubicado en la guía de trabajo, es decir, indirectamente en el profesor. Mientras que bajo la metodología orientada a la cooperación, este rol principal es cedido al estudiante como centro del proceso de aprendizaje y el cumplimiento del objetivo/desafío como la prueba irrefutable del conocimiento adquirido.

El cambio del método tradicional al orientado a la cooperación, trajo consigo que los estudiantes actuaran en forma grupal y realizaran un trabajo colaborativo, integrando sus conocimientos (adquiridos aparentemente en forma aislada), creando sus propias estrategias de resolución de problemas, planificando previamente los pasos a seguir para abordarlos, resolviendo dudas acerca del enunciado, homologando criterios y representando la o las posibles soluciones. Además permitió fortalecer y desarrollar habilidades tales como liderazgo, capacidad de trabajo en equipo, razonamiento crítico, autonomía y autoaprendizaje, todas ellas muy valoradas por el mercado laboral y por los estudios de programas curriculares para ingeniería. Todo esto hizo que los alumnos sintieran y manifestaran mayor seguridad en sus conocimientos sobre la materia.

Hay que decir que el método recibió la aprobación de todos los integrantes del curso en forma unánime lo que lo validó también ante ellos haciendo que apreciaran los cambios de una forma positiva.

Por lo tanto, se puede concluir que al aplicar una metodología como la que se presentó, es posible observar una experiencia exitosa de enseñanza.

Referencias

[1] Ander-Egg, Ezequiel. DICCIONARIO DE PEDAGOGÍA. 1ª edición. Buenos Aires, Argentina. 1997.
[2] Instituto Tecnológico y de Estudios Superiores de Monterrey. EL APRENDIZAJE BASADO EN PROBLEMAS COMO TÉCNICA DIDÁCTICA, http://www.sistema.itesm.mx/va/dide/inf-doc/estrategias. 2000.
[3] Cataldo, Alejandro – Aliaga, Verónica. METODOLOGÍAS INNOVADORAS DE ENSEÑANZA Y CENTRADAS EN EL ALUMNO EN LA CARRERA DE INGENIERÍA INFORMÁTICA EN LA UNIVERSIDAD DE ATACAMA. (Por editar). I Simposio Iberoamericano de Educación, Cibernética e Informática (SIECI 2004). Orlando, EE.UU. 2004.
[4] Aliaga, Verónica. INNOVACIONES EN PEDAGOGÍA UNIVERSITARIA. V Encuentro Nacional de Psicología Cognitiva. Santiago, Chile. 2003.
[5] Lorenz, Jim. Cisco Systems Inc. ACADEMIA DE NETWORKING DE CISCO SYSTEM - PRÁCTICAS DE LABORATORIO (VOLUMEN I). 2ª edición. Pearson Educación. Madrid, España. 2002.

Elaboración de material educativo para la formación de profesionales en desarrollo de software

Edgar E. Casasola

Universidad de Costa Rica, Escuela de Ciencias de la Computación e Informática
San Pedro, Costa Rica 2060
ecasasol@ecci.ucr.ac.cr

Abstract

This paper documents the first country wide experience related to collaboration between academy, industry and government, towards the development of educational material used to improve the quality of software development professionals. This effort is part of the educational component of the PROSOFTWARE project. The goal of the project is to strength the Costa Rican software development enterprises. This paper describes the context, background, methodology and results. It resumes the results obtained after finishing the first of a series of two courses on computer programming. This paper could be useful for those individuals or organizations interested on course design and implementation towards the education of well formed software development professionals.

Keywords: course planning, teaching and learning support environments, teaching methodology

Resumen

Este artículo documenta la primera experiencia a nivel nacional de colaboración entre academia, industria y gobierno asociada a la elaboración de un material educativo para el mejoramiento de la formación de profesionales en desarrollo de software. Este esfuerzo corresponde a resultados concretos del componente educativo del proyecto PRO-SOFTWARE. Proyecto creado con el fin de fortalecer a la industria de desarrollo de software en Costa Rica. Se describen el contexto y antecedentes del trabajo, la metodología de desarrollo, y la descripción de los resultados obtenidos durante la elaboración exitosa del primer material correspondiente a una serie de dos cursos de programación. Este artículo puede ser de interés para aquellos individuos u organizaciones interesados en la sistematización de cursos para la formación de profesionales idóneos para el desarrollo de software.

Palabras clave: planes de estudio, ambientes de apoyo a la enseñanza, metodología de enseñanza

1. Introducción

El presente trabajo es parte de los resultados obtenidos luego de desarrollar un material interinstitucional para impartir el primer curso de programación de computadoras. El material está diseñado para ser utilizado por los principales centros de enseñanza a nivel universitario en Costa Rica. Este curso es un producto directo del componente educativo del proyecto PROSOFTWARE, una iniciativa de la Cámara de Productores de Software de Costa Rica CAPROSOFT [4]. CAPROSOFT, fundada en 1998, es un consorcio sin fines de lucro formado por 60 compañías de desarrollo en este sector. Una de sus primeras iniciativas fue crear PROSOFTWARE, un proyecto conjunto entre la industria del software, el gobierno y las universidades para mejorar la competitividad de sus miembros mediante el mejoramiento de la calidad.

El proyecto PROSOFTWARE está constituido por tres componentes principales tal y como se puede observar en la Figura 1.

Este trabajo se ubica en el componente educacional del proyecto. Este componente persigue "la formación de profesionales idóneos", donde la idoneidad del recurso humano es vista como "la capacidad y habilidad de los graduados para desempeñarse de manera óptima de acuerdo con las necesidades y demandas del entorno" [7].

Dentro del contexto del componente educacional se llevaron a cabo tanto un Estudio de Oferta Demanda de Profesionales en el sector de Desarrollo de Software, como un Estudio para el Fortalecimiento de los Centros de Enseñanza y Actualización Curricular. El estudio de oferta y demanda de profesionales en el sector software se llevó a cabo entre octubre del 2000 y junio del 2001 con la participación de 97 empresas desarrolladoras de software de 124 identificadas, 53 organizaciones de otros sectores, y 123 profesionales. Y en octubre del 2003 se concluyó el Estudio de Perfiles Profesionales y Académicos. Este estudio, dividido en fases, implicaba la

identificación de perfiles de desempeño ocupacional, perfiles académicos-profesionales y finalmente la elaboración de recomendaciones curriculares con diseño de mallas curriculares propuestas según el perfil. [7]

Con el fin de promover aún más el fortalecimiento de los centros de enseñanza a nivel superior se procedió a identificar los cursos considerados medulares o cursos pertenecientes a un eje común para todos los perfiles. Se determinó que el primer curso de programación es fundamental para la formación de la mayoría de los profesionales en la industria del software y se tomó la decisión de contratar un coordinador de contenido para encargarse de la sistematización de este primer curso y coordinar esfuerzos entre profesores universitarios de todo el país.

Este documento presenta los resultados de la culminación exitosa del diseño del primer curso de Programación, el cual forma parte de una serie de dos. La versión oficial y validada del primer curso se utilizará oficialmente a partir de julio del 2004. Se espera contar con la primera versión del segundo curso para agosto del 2004.

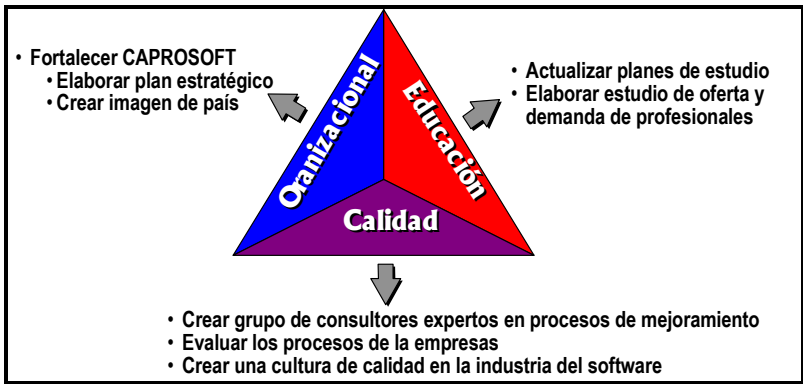


Figura 1. Los tres componentes principales del proyecto PRO-SOFTWARE [6].

2. Metodología utilizada

El trabajo de gestión de esta etapa del proyecto inició en paralelo con la presentación de los resultados preliminares obtenidos como parte de los estudios presentados por Mata y Matarrita [7]. Este trabajo de gestión se llevó a cabo ante los representantes de los centros de estudio participantes. En este momento se solicitó la participación de profesores y se propuso la estructura de trabajo que se muestra en la Figura 2.

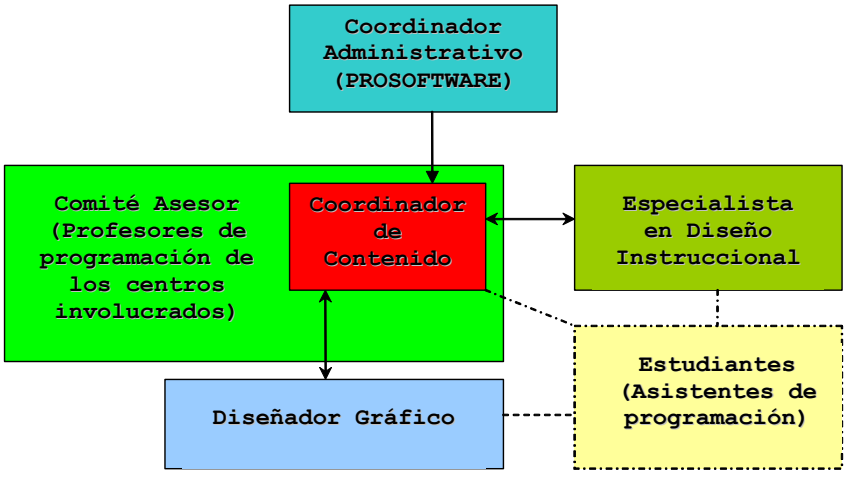


Figura 2. Conformación del equipo de trabajo.

Como se muestra en la Figura 2, el comité asesor estaría formado de profesores de programación representantes de cada centro, su trabajo estaría dirigido por el coordinador de contenido con el fin de definir las características deseables para un curso que fuera de utilidad para todos los involucrados. Todos los aspectos administrativos los manejaría un representante del proyecto Pro-Software dejando los aspectos relacionados al

contenido del curso bajo la responsabilidad del coordinador respectivo. El grupo de profesores fue entonces el generador de las características deseables del curso, utilizando los dos insumos básicos que fueron los planes y experiencia de los docentes de todos los centros y el conocimiento de los perfiles académicos y profesionales y de las recomendaciones curriculares propuestas en los estudios previos.

El trabajo se llevó a cabo bajo la modalidad de grupos focales, donde el coordinador se encargaba de propiciar una discusión que estimulaba a los participantes a compartir sus percepciones e ideas sobre un determinado tópico lo cual genera información a fondo sobre los temas en cuestión. Esta técnica propició motivar e identificar las características y necesidades o problemáticas comunes entre los diferentes centros.

La primera etapa del trabajo con el comité asesor fue la discusión y análisis de los planes de estudio actuales de los diferentes centros. Cabe mencionar que la principal preocupación se orientó hacia la selección del paradigma de programación y el lenguaje que se utilizarían.

En sesiones posteriores se discutió una propuesta de plan de curso y la discusión giró alrededor de los objetivos del mismo, contenidos, la modalidad de curso y las características que debía cumplir un material que fuera útil para todos los centros participantes.

Con el insumo anterior se preparó la primera versión del material y se llevó a cabo un taller de presentación, análisis y retroalimentación con el grupo asesor y un profesor invitado extra por centro. En este taller se resaltaron los aspectos positivos y negativos de la primera versión del material, y dichos insumos vinieron a propiciar la base para la elaboración de la versión 2 del material que incorpora todas las mejoras discutidas.

En la siguiente sección se presentan los resultados obtenidos con ejemplos concretos del curso implementado.

3. Resultados concretos

A. Análisis de los planes de estudio de los diferentes centros

En todos los centros se detectó alguna deficiencia relacionada con el diseño instruccional. En la mayoría de los centros: la sistematización de los cursos no era total, estaba orientada solamente a una lista de contenidos sin especificación del nivel al que se debía tratar cada tema, el perfil de salida del estudiante dependía en gran medida de la "interpretación" que haga el profesor del plan existente. En la mayoría de los casos simplemente se confiaba en la experiencia de los profesores asignados para impartir los cursos.

El primer resultado positivo fue resaltar la importancia de la sistematización de los cursos, y que esto llevaría a la minimización del esfuerzo en cuanto a planeamiento, programación de ejemplos y ejercicios. Por otra parte el aseguramiento de un nivel mínimo de calidad, y la posibilidad de que el estudiante avance a su propio ritmo.

Otro punto importante fue la definición de las habilidades fundamentales que se desea que el estudiante desarrolle para llegar a ser un buen profesional. Y la conclusión de que aspectos que no son centrales para el primer curso de programación pero que son importantes para la formación de un profesional en desarrollo de software deben incorporarse en forma temprana como "ejes transversales" cuyo abordaje se incrementará conforme el estudiante avance en su plan de estudios. Por ejemplo el conocimiento de la importancia de la comunicación oral y escrita, trabajo en grupo y la introducción de aspectos éticos relacionados al desarrollo de software se consideraron ejes transversales. Se definió la importancia de contar con pequeños proyectos elaborados en grupos de varios estudiantes con períodos de tres a cuatro semanas de duración, a los cuales se les denomina en el medio como "tareas programadas". El producto específico asociado a la calidad de la documentación, fue la definición de un estándar para la presentación de estas "tareas programadas" y el uso de generadores de documentación de código fuente como el javadoc [9].

B. Importancia de la selección del paradigma de programación

La importancia de la selección del primer paradigma de programación siempre fue tema de discusión, algo que se manifiesta en la gran cantidad de artículos existentes sobre el tema, por ejemplo estudios como el de Zhu, Haibin y Zhou [10]. Por lo tanto, la selección de paradigma de programación, y de los posibles lenguajes para la enseñanza del mismo fueron definidos desde el inicio.

Se tomaron en cuenta las recomendaciones curriculares del conjunto formado por la ACM, IEEE-CS y AIS conocido como La Fuerza Conjunta para el Currículo Computacional o "Joint Task Force for Computing Curricula 2004" [1] y los resultados del estudio de perfiles profesionales y académicos con que se contaba. Este último estudio vino a reforzar la importancia de contar con profesionales que dominaran el desarrollo de software orientado a objetos, modular, siguiendo patrones de diseño y preferiblemente en capas múltiples con interfaz Web. El sector software consideró importante el conocimiento en cuanto a las tecnologías emergentes en el mercado nacional incluyendo .NET de la corporación Microsoft [8] y J2EE de Sun Microsystems [9].

C. Definición de los objetivos del curso

El apoyo del diseñador curricular fue importante para la definición de los objetivos del curso. Se mencionó que el verbo utilizado para la redacción de cada objetivo llevaría implícito el nivel de aprendizaje esperado. Se incorporó al trabajo el uso de la una adaptación de la escala de Bloom [3], presentada en forma de pirámide y la cual se muestra en la Figura 3.

Esta escala indica que un objetivo con uno de los verbos colocados a un nivel de la pirámide implica el cumplimiento de objetivos en cada uno de los niveles inferiores. Por ejemplo el verbo "fundamentar" incluye aprendizaje asociado a los niveles inferiores por ejemplo (ser capaz de identificar, distinguir, ejemplificar y categorizar).

El plan del curso incluye la definición de un objetivo general y el detalle de los objetivos específicos. El objetivo general del curso quedó redactado de la siguiente forma:

> **Al terminar el curso el estudiante será capaz de resolver problemas simples mediante el diseño de algoritmos y desarrollo de programas, aplicando técnicas actuales de desarrollo de software orientado a objetos y considerando criterios de calidad apropiados.**

Como objetivos específicos se espera que al finalizar el curso el estudiante sea capaz de:

- Identificar problemas específicos.
- Representar problemas mediante la utilización de modelos abstractos.
- Comprender los elementos y estructuras básicas presentes en un lenguaje de programación orientado a objetos para implementar programas modulares, claros, simples y generales.
- Aplicar a nivel básico buenas prácticas de construcción de software tales como uso de estándares de documentación, codificación, verificación y validación para el aseguramiento de la calidad.
- Analizar problemas mediante un proceso de descomposición y refinamiento en pasos sucesivos.
- Sintetizar los resultados del proceso de análisis para diseñar algoritmos para la resolución de problemas.
- Aplicar una metodología de resolución de problemas que le permita trabajar con orden y disciplina.
- Descubrir la importancia del desarrollo de buenas prácticas de trabajo en equipo y de comunicación oral para la resolución de problemas.
- Aplicar técnicas apropiadas de comunicación escrita para la documentación de la solución de cada problema.

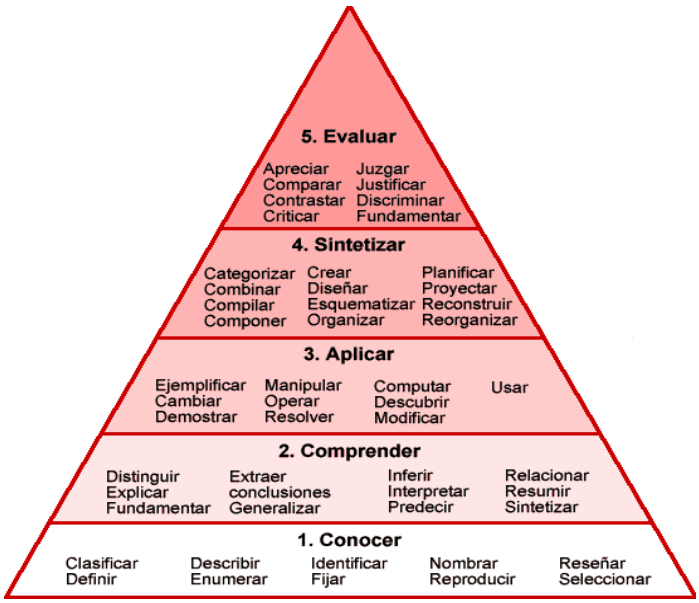


Figura 3. Diagrama que muestra los verbos utilizados para la definición de los objetivos de aprendizaje, organizados según la taxonomía de Bloom.

D. El lenguaje de programación

Java fue seleccionado por múltiples razones. Principalmente por ser un lenguaje independiente de la plataforma tecnológica de cada centro y cuyo uso no obliga a los centros a contar con licencias de alto costo. Otra

característica particular es la que algunos centros de estudio con programas de diplomado no cuentan con el tiempo suficiente dentro de sus programas para enseñar varios lenguajes de programación por lo que con Java pueden utilizar un lenguaje que también tiene utilidad para cursos de ingeniería de software y en el mercado laboral les permitirá el desarrollo de aplicaciones empresariales. Según se mencionó en el estudio previo se considera importante en este medio el conocimiento de tecnologías como la Edición Empresarial de Java 2 o J2EE [9] utilizadas para desarrollo de software en múltiples capas y aplicaciones Web. Se discutió que la mayoría de los estudiantes acostumbran trabajar en sus casas por lo que utilizar herramientas de alto costo implica obligar a los estudiantes a trasladarse a los centros para llevar acabo sus trabajos. Se consideró además que el uso de java evita que los estudiantes desarrollen el hábito de instalar software sin licencia en sus computadores de uso personal alegando que el costo de las licencias está fuera de su alcance, la formación puede ser integral si se presentan alternativas menos costosas para el estudiante.

E. Compilador y ambiente de desarrollo de programas

Al referirse al compilador y ambiente de desarrollo más recomendable se discutió la importancia de que el estudiante sea capaz de distinguir y diferenciar conceptos como lenguaje, compilador y herramienta de desarrollo. Además, que pueda distinguir las fases del proceso de escritura, compilación y ejecución de programas. Se mencionó la importancia de introducir la compilación mediante comandos para comprender los pasos que llevan a cabo las herramientas de desarrollo.

Al inicio del curso el estudiante aprende a distinguir las fases del proceso de creación, compilación y ejecución de un programa tal y como se muestra en la figura 4. En los primeros ejercicios se aprende el proceso de compilación y ejecución paso a paso mediante el uso de comandos como "javac" y "java" pertenecientes a las herramientas de desarrollo en Java o jdk de Sun Microsystems [9]. Por otra parte se introduce el uso de ambientes integrados de desarrollo como el Dr. Java [2] marcando la diferencia entre los conceptos "lenguaje", "compilador" y "ambiente integrado de desarrollo" o "IDE" según sus siglas en inglés. Dr.Java deja de lado toda funcionalidad innecesaria en un curso básico de programación.

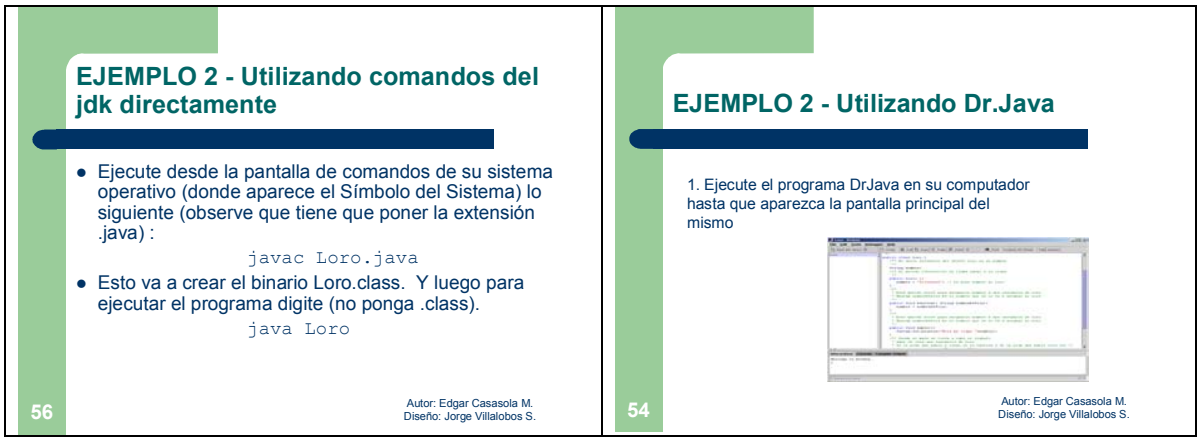


Figura 4. En los primeros ejemplos el estudiante se familiariza con diferentes formas de compilar y ejecutar un programa.

Otra característica importante fue la importancia de que se diera énfasis en los conceptos y no en el lenguaje seleccionado por lo que en todo el curso se mantiene una separación entre el contenido y la implementación en el lenguaje específico. Esto se manifiesta en el material con la separación explícita tal y como se muestra en la figura 5. Note que cada concepto presenta por separado los aspecto de implementación logrando que una posible actualización o expansión del material utilizando otros lenguajes se pueda dar de manera natural, facilitando el mantenimiento del curso.

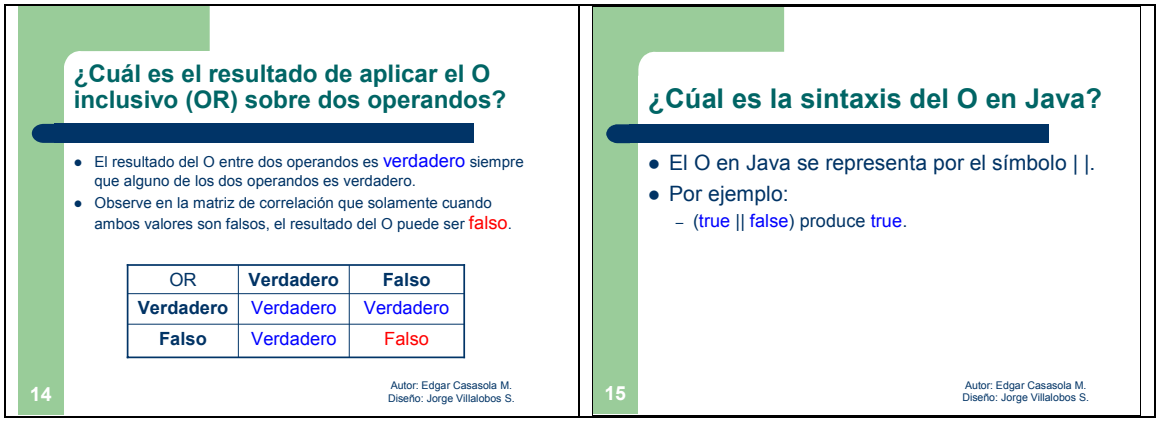


Figura 5. Separación explícita entre la sintaxis del lenguaje y los conceptos de programación en el material que presenta los contenidos del curso.

F. Aspectos curriculares asociados al contenido

Se discutió la importancia de no dar tanta importancia a la cantidad de contenidos como a las habilidades que se desarrollen en el estudiante, además de dar énfasis a la comprensión de los procesos generales llevándolos posteriormente a su representación sintáctica. En este caso se recurrió al uso de animaciones paso a paso como se muestra en la figura 6. Por ejemplo: Visualización de la memoria (Pila, memoria estática y el montículo principal o "Heap") durante la construcción y destrucción de objetos, mecánica de los llamados a métodos, recursividad, recorridos en vectores y recorridos en matrices.

Se discutió si la existencia de cursos con la modalidad de laboratorio para práctica dirigida era apropiado para los cursos de programación o si era más recomendable la creación de cursos bi-modales donde el estudiante pudiera experimentar a su propio ritmo fuera del aula y disponga del instructor para aclarar dudas y profundizar en la resolución de casos y llevando a cabo un rol de tutor que se encarga de aclarar conceptos. Fue importante la intervención del experto en diseño curricular quien fundamentó las virtudes de los modelos de autoaprendizaje, educación a distancia y educación asistida por computador. Por lo tanto se requería de un curso que tendiera más a enfoque menos presencial pero que permitiera el uso del material y contenidos en el aula.

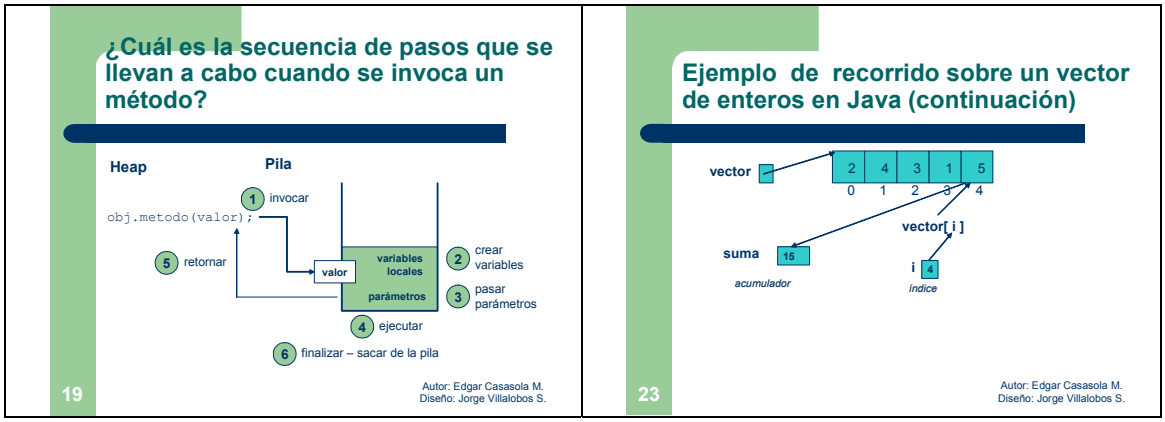


Figura 6. Fue necesario ilustrar procesos conceptualmente importantes mediante el uso de animaciones.

G. Estructura global del material y presentación del curso

Durante las discusiones con el grupo asesor se detectó una tendencia casi generalizada hacia el enfoque magistral y la idea del profesor como generador de conocimiento y el estudiante es un mero receptor de información y no como el eje central del proceso de aprendizaje. Se pensó entonces que para la transición hacia enfoques menos magisteriales el material podría utilizar un enfoque bi-modal con definición de horas presenciales y horas extra clase. El material debía ser suficientemente versátil como para poder ser utilizado en el aula como de manera individual por el estudiante.

El curso mantiene una estructura navegable y permite acceder directamente desde el árbol de la parte izquierda a las partes. El mismo fue desarrollado con una interfaz Web mediante el uso de html y javascript sobre

un modelo de datos reducido que mantuviera las mismas características de presentación a través de diferentes navegadores, lo cual unido al uso de Java ofrece la opción de ser multiplataforma e independiente de la tecnología utilizada en cada centro. Permitiendo su presentación y navegación tanto a través del Web como directamente desde un CD si así se desea.

La Figura 7 muestra en ejemplo de presentación de contenidos. Al lado izquierdo se presenta un árbol con la estructura del curso la cual permite navegar por los diferentes componentes del curso. Al lado derecho aparece un ejemplo de una pantalla de contenido. El estudiante puede repasar contenidos específicos a su propio ritmo. Y el docente puede maximizar la pantalla de presentación en clase. El formato de presentación es apto para ser utilizado en video conferencias.

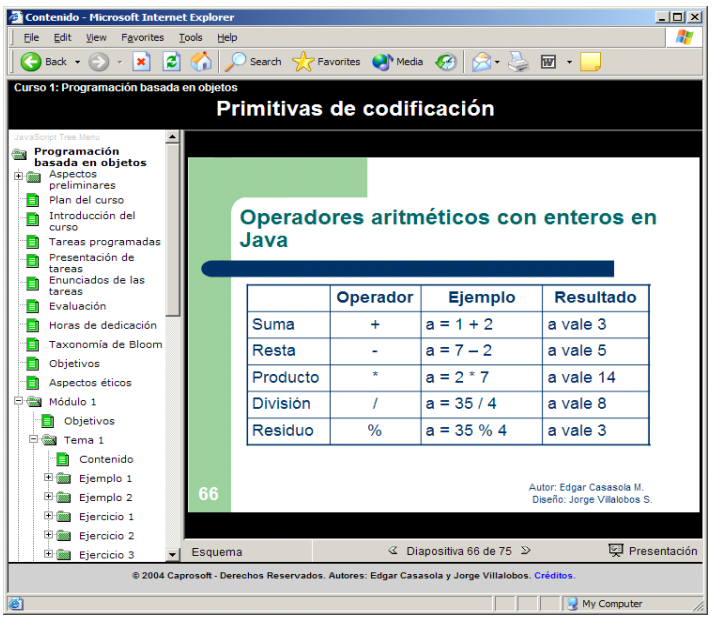


Figura 7. Presentación de los contenidos del curso.

De las sesiones de trabajo con el grupo asesor surgió la opinión generalizada de la necesidad de que el estudiante se mantenga siempre activo en cuanto a programación. El estudiante aprende mejor cuando siente que necesita del conocimiento para lograr algo, como por ejemplo resolver algún problema que le aqueja. De este modo se resaltó la figura de los proyectos o "tareas de programación" donde el estudiante se enfrenta a un problema que debe ser resuelto en grupo a lo largo de tres o cuatro semanas de trabajo y donde la solución del mismo no puede ser visualizada de forma inmediata como en ejercicios pequeños tendientes a reforzar conceptos. Esta modalidad se presta para que los estudiantes puedan desarrollar habilidades "transversales" como el trabajo en equipo y la comunicación oral y escrita, mediante la presentación de reportes y documentación de proyectos. Además permite llevar a cabo proyectos de mayor nivel supervisados por su instructor y es apta para todos los centros. El ejercicio en cuanto al proceso de identificación del problema, análisis del problema, diseño de una solución, implementación de la solución y prueba del mismo es reforzado a mayor escala durante el desarrollo de las tareas programadas.

A diferencia de otros cursos donde todo se centra en el código fuente, en este se presentan los ejemplos y ejercicios separados en las etapas del ciclo de resolución de un problema tal y como se muestra en la figura 8. El estudiante y el profesor cuentan con copias que se diferencian en las instrucciones incluidas. Además la versión del profesor tiene soluciones para todos los ejercicios incluyendo su análisis, diseño, implementación y posibles pruebas. En el caso del estudiante y dependiendo del enunciado del problema puede que en algunas de estas secciones lo que exista sean instrucciones que le sirvan de guía para llevar a cabo esa etapa en forma independiente. Ambas versiones permiten utilizar el material discusión en clases magistrales mediante la maximización de las presentaciones de contenido.

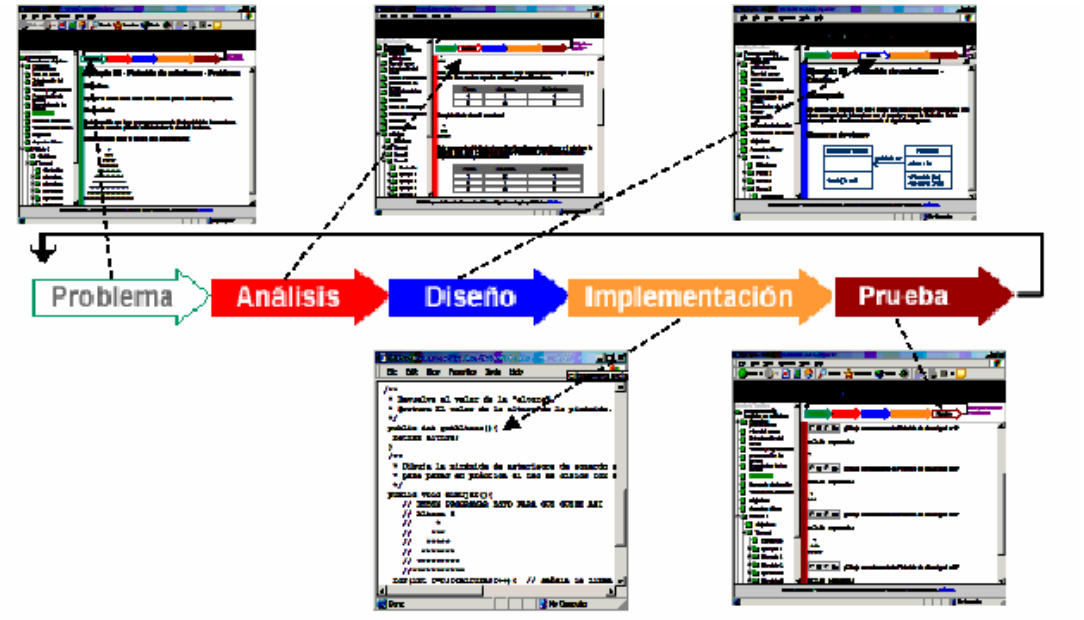


Figura 8. Ejemplos y ejercicios dan énfasis a la resolución de problemas.

De igual forma se hace énfasis en aspectos como la buena especificación y documentación de programas, y en la importancia del diseño de pruebas y el uso de listas de chequeo. Se quiere dejar claro en todo momento que aprender a programar no implica solo generar código, lo cual puede ser traumático para algunos si se trata de llevar a cabo de forma directa, sino ir de la especificación del problema hasta una solución probada siguiendo una metodología apropiada., donde en cada etapa hay un producto que sirve como insumo para la etapa siguiente.

H. Documentación de programas

Finalmente se mencionan dos aspectos o soluciones para el tema de calidad en el desarrollo de software. En este caso nos referimos a documentación y especificación de programas. En este caso la solución propuesta se centró en el la definición de un estándar simple para documentación de tareas y en el uso de especificación del código fuente mediante javadoc para llevar a cabo generación automática de documentación.

Un ejemplo de la documentación generada se puede visualizar en la Figura 9. En el curso se introduce el uso de comentarios dentro de los programas utilizando javadoc para la documentación apropiada de los programas.

I. Sistematización de criterios de evaluación

Por último es importante que tanto el estudiante como el docente tengan claro cuales son los aspectos que se evaluarán y la ponderación o importancia que se asigna a cada uno. La sistematización de este proceso se llevó a cabo mediante plantillas de evaluación con sus respectivas instrucciones de aplicación. Un ejemplo de plantilla de evaluación de tareas se muestra en la figura 9. Se sugieren algunas plantillas de evaluación de tareas cortas para que tanto el instructor como el estudiante tengan claro el énfasis que se da a cada aspecto de las soluciones planteadas. Las plantillas pretenden promover la sistematización de los mecanismos de evaluación con el fin de motivar y capacitar a los nuevos docentes en aspectos relacionados.

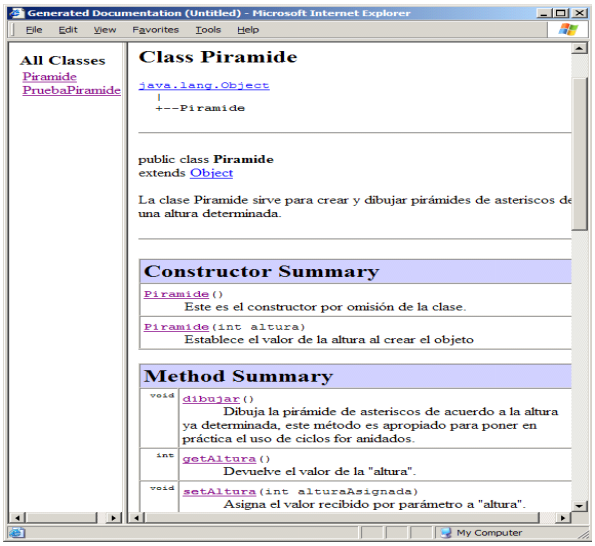


Figura 9. Uso de Javadoc.

4. Conclusiones

En Costa Rica es la primera vez que se sistematiza un curso para la formación de profesionales en desarrollo de software a nivel universitario con colaboración de todos los centros de enseñanza superior del país. El curso permite trazar la línea curricular originada en el estudio de oferta y demanda de profesionales para el desarrollo de software [7], hasta llegar a los contenidos, ejemplos y ejercicios del curso. Todos los ejemplos y ejercicios cuentan con objetivos explícitos que los relacionan con el contenido. El contenido satisface uno o más objetivos del curso lo cual demuestra una alta coherencia en el diseño instruccional del mismo. Todo esto facilita al estudiante y al docente para ubicarse y tener en todo momento claro: ¿que se quiere?, ¿para qué se quiere? y ¿cómo se quiere?

EVALUACION DE TAREA						
	MUY ALTO	ALTO	REGULAR	BAJO	MUY BAJO	NOTA
DOCUMENTACION INTERNA						
Tabulación y anidamiento correcto	8	6	4	2	0	
Nombres significativos	8	6	4	2	0	
Usa comentarios para especificar variables, métodos, parámetros y clases	8	6	4	2	0	
Consistencia en uso de convenciones de programación	6	4	2	1	0	
LOGICA DEL CODIGO FUENTE						
Claro	15	10	6	3	1	
Modular	15	10	6	3	1	
EJECUCION						
Compila y ejecuta sin errores de sintaxis	10	6	3	1	0	
Soluciona correctamente el problema planteado	15	10	6	3	1	
Casos de prueba apropiados	15	12	9	6	3	
FACTOR DE COMPLETITUD	x 1	x 0.8	x 0.6	x 0.4	x 0.2	
TOTAL						

Figura 10. Uso de plantillas de evaluación.

El diseño bi-modal del curso es apto tanto para estudiantes de los centros que utilizan un enfoque de enseñanza a distancia acompañado de tutorías, como para centros con enfoques presenciales. La portabilidad y compatibilidad del material es adecuada ya que no depende de plataformas comerciales para la enseñanza y para trasladarlo tanto en CD como el acceso vía Web. La versatilidad del material permite que el mismo sea utilizado para presentación de material y discusión bajo el enfoque presencial en el aula, como para auto aprendizaje por parte del estudiante.

El uso de las tareas programadas como disparador del trabajo a lo largo de todo el curso es importante para mantener al estudiante activo durante todo el proceso de aprendizaje y para que aplique e integre los conceptos puntuales vistos en el contenido. El curso logra enfocarse en aspecto como resolución de problemas más que enseñanza de un lenguaje de programación. En el mismo se separan claramente los conceptos de los aspectos de implementación en Java por lo que puede extenderse fácilmente con ejemplos en lenguajes como C++ o C# si así se deseara. Razón por la que la obsolescencia del curso será más lenta.

Se espera que el uso apropiado del material y la sistematización de la evaluación ayuden a asegurar un nivel de aprendizaje idóneo, y homogeneidad entre los estudiantes que aprueban el curso. Lo anterior facilitaría el proceso de articulación con los siguientes cursos de la carrera y finalmente formar profesionales idóneos para el medio en que se desempeñan.

5. Trabajo Futuro

Actualmente se está trabajando en la elaboración de un segundo curso de programación como continuación del primero para introducir aspectos como mecanismos de reutilización, programación por eventos, flujos de datos, hilos, y comunicaciones. Se plantea la necesidad de capacitar profesores en cuanto al uso apropiado del paradigma orientado a objetos y la necesidad de completar el esfuerzo mediante la creación de un curso compartido de estructuras de datos y análisis de algoritmos, otros dos de ingeniería de software y otro en diseño de bases de datos para completar el núcleo común. Los profesores participantes de cada centro han abierto un canal de colaboración que se planea utilizar para llevar a cabo al menos un taller anual de discusión para el mejoramiento continuo de la enseñanza de la programación, y se plantea contar con un foro de discusión continua entre centros. Actualmente se discute con los personeros de la Cámara de Productores de Software sobre la posibilidad de financiar un nuevo proyecto tendiente a evaluar a largo plazo el aprovechamiento por parte de los docentes al utilizar el material, y el impacto en el aprendizaje de los estudiantes.

Agradecimientos: MBa. Adolfo Cruz (Director ejecutivo Prosoftware), MSc. Eduardo Araya (Coordinador administrativo Prosoftware), Dr. Francisco Mata, profesores universitarios de las instituciones participantes (CENFOTEC, CUC, CUNA, CUP, ITCR, UNED, UCR, UCR, ULATINA, UNA), Bach. Jorge Villalobos (diseñador gráfico), MSc. Melvin Chaves (diseñador curricular), y a los estudiantes Adriana Blanco y Francisco Villegas.

6. Referencias

[1] Joint Task Force for Computing Curricula. **Computing Curricula 2004-Overview Report.** Association for Computing Machinery (ACM), Association for Information Systems (AIS), y Computer Society (IEEE-CS). 2004. http://www.acm.org/education/curricula.html

[2] Allen, E.; Cartwright, R.; y Stoler, B. DrJava: A lightweight pedagogic environment for Java. En: **33rd ACM Technical Symposium on Computer Science Education (SIGCSE 2002)**, Northern Kentucky, Cincinnati, USA, February 27 - March 3, 2002.

[3] Bloom, B. (1956). **Taxonomy of educational objectives. Handbook I. The cognitive domain**. New York, David McKay & Co., 1956.

[4] Caprosoft . **Estudio de Oferta y Demanda del Recurso Humano en el Sector Software Costarricense.** [página principal WWW], Visitada: Junio 2004. http://www.caprosoft.org/Caprosoft - Proyecto BID.htm

[5] Duke, R.; Salzman, E.; Burmeister,J.; Poon, J.; y Murray, L. Teaching programming to beginners - choosing the language is just the first step. En: **Proceedings of the Australasian Conference on Computing Education. Melbourne,** Australia, 2000.

[6] Jenkins, M. PRO-SOFTWARE: A Government-Industry-Academia Partnership that Worked. En: **17th Conference on Software Engineering Education and Training (CSEET'04).** Norfolk, Virginia. 01 – 03 March, pp. 92-97, 2003

[7] Mata, F.; y Matarrita, R.. **Conclusiones y Recomendaciones del Estudio para el Fortalecimiento de los Centros de Enseñanza y la Actualización Curricular.** CAPROSOFT. San José, Costa Rica. 2003 http://www.caprosoft.org/publicaciones.shtml

[8] Microsoft, **Página web**. Julio 2004. http://www.microsoft.com/

[9] Sun Microsystems. **Página web.** Julio 2004. http://www.sun.com/

[10] Zhu, H.; Zhou, M. Methodology first and language second: a way to teach object-oriented programming. En: **18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications.** Anaheim, California, USA, 2003.

Tesis de Maestría
Master's Thesis

Resolución con orden y selección para la lógica $\mathcal{H}(@)$

Daniel Alejandro Gorín
Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires
Ciudad Universitaria – Pabellón I (1428) Buenos Aires, Argentina

Director: Carlos Eduardo Areces
INRIA Lorraine
615, rue du Jardin Botanique 54602 Villers les Nancy Cedex, France

Resumen

Las lógicas modales son reconocidas por combinar buenas propiedades computacionales (e.g. decidibilidad) con un alto poder expresivo. Estas propiedades han sido explotadas con éxito en Computación; un caso paradigmático lo constituyen las *description logics* (DLs), una familia de lógicas modales utilizadas en la construcción de *sistemas de representación de conocimiento* (knowledge representation systems o KRS). En este tipo de sistemas se puede dar una descripción conceptual de un *universo* junto con una enumeración de los elementos que lo constituyen, y a partir de ahí, inferir información que no se encuentra declarada en forma explícita. Este tipo de sistemas se utilizan, por ejemplo, en Inteligencia Artificial y en Lingüística Computacional, y han sido propuestos como base para el lenguaje de representación de ontologías que se usaría en la Web Semántica [4].

Las lógicas modales tradicionales, sin embargo, no permiten hacer referencia a elementos particulares del modelo, ni permiten tampoco representar igualdades. La lógica híbrida $\mathcal{H}(@)$ es la que se obtiene al agregar nominales y el operador de satisfacción @ a la lógica modal básica. Los nominales permiten *nombrar* elementos del modelo y junto con el operador @ incorporan una noción débil de igualdad. A pesar de su mayor poder expresivo, $\mathcal{H}(@)$ sigue siendo decidible: de hecho, su complejidad para el problema de la satisfaciblidad es igual al de la lógica modal básica, PSPACE-completo.

Habitualmente, los demostradores de teoremas para las lógicas modales están basados en algoritmos de tableau que, combinados con diversas heurísticas y optimizaciones, muestran buen comportamiento empírico. Sin embargo, muchas de estas heurísticas no son correctas cuando la lógica incorpora igualdad. Es interesante, entonces, investigar qué sucede con otras familias de algoritmos.

En [2] se propone un cálculo basado en resolución para $\mathcal{H}(@)$. Este cálculo, si bien consistente y completo, carece de las estrategias de orden y selección que son parte esencial de los demostradores para lógica de primer orden basados en resolución. Una implementación computacionalmente realista de un demostrador basado en resolución requiere de este tipo de estrategias para regular la generación desproporcionada de nuevas cláusulas y como guía dentro de un espacio de búsqueda extremadamente complejo.

El primer resultado de esta tesis es la definición de una estrategia de orden y selección muy general para el cálculo de resolución propuesto en [2], que preserva completitud refutacional (i.e., si bien el espacio de búsqueda disminuye drásticamente, el algoritmo de refutación sigue siendo correcto y completo). Como un paso necesario para la demostración general de completitud, se presenta un Teorema de Herbrand [11] para la lógica $\mathcal{H}(@)$. Este resultado permite reducir el problema de satisfacibilidad sobre la clase de todos los modelos posibles a la subclase de los modelos de Herbrand (este es, hasta donde sabemos, el primer resultado de este tipo para lógicas modales).

El segundo resultado que la tesis presenta es una demostración de terminación para el cálculo propuesto previamente. Tomando ventaja de la generalidad de la demostración anterior, podemos modificar el cálculo para asegurar terminación dada cualquier entrada. Si bien en [1] se demuestra que la complejidad del problema de satisfacibilidad para $\mathcal{H}(@)$ es PSPACE-completo (y por lo tanto decidible), nuestra demostración provee el primer algoritmo computacionalmente realizable.

Por último, los resultados teóricos de esta tesis fueron puestos a prueba en la re-implementación del prototipo HyLoRes, un demostrador automático basado en el cálculo de [2]. Los tests que presentamos muestran claramente que las estrategias de orden y selección también producen una mejora drástica en un algoritmo de resolución para lógicas híbridas, obteniendo tiempos de cómputo varios órdenes de magnitud menores.

1. Introducción

1.1. Lógicas modales e híbridas

Las lógicas modales proposicionales fueron propuestas para formalizar conceptos tales como posibilidad, obligación, conocimiento, etc. Sin embargo, en la década del '60, trabajos como [12, 13, 14] mostraron que estas lógicas también podían interpretarse usando ciertas estructuras relacionales, hoy llamadas *modelos de Kripke*. Estas estructuras se pueden ver como multigrafos dirigidos, donde los nodos se etiquetan con el conjunto de todas las proposiciones que valen en cada uno. A partir de ese momento, se amplió considerablemente el campo de aplicación de las lógicas modales; en particular se encontró que eran de gran utilidad en diversas disciplinas de Computación.

La ventaja de las lógicas modales en este terreno sobre otras lógicas clásicas está en que presentan un poder expresivo relativamente alto, sin perder decidibilidad: el problema de determinar si una fórmula de la lógica modal básica es *válida* (o, dualmente, *satisfacible*) es PSPACE-completo. Si bien con esta complejidad de peor caso este problema cae dentro de los denominados *intratables*, pruebas empíricas muestran que para un número importante de instancias de utilidad práctica, demostradores automáticos pueden dar una respuesta en tiempos razonables.

A pesar de su alto poder expresivo, las lógicas modales tradicionales presentan algunas limitaciones importantes, e.g. no permiten hacer referencia explícita a elementos concretos del dominio ni hablar de igualdad entre elementos. Las *lógicas híbridas* son una familia de extensiones de la lógica modal clásica que intentan resolver esta limitación mediante la introducción de *nominales* y operadores modales especiales.

Intuitivamente, un nominal es un *nombre único* para un elemento del modelo. Desde un punto de vista sintáctico, un nominal es similar a un símbolo de proposición, y puede ser usado en cualquier contexto donde éste sea aceptable. Por ejemplo, si i y j son nominales, y p es un símbolo de proposición, podemos escribir fórmulas como $i \wedge p \wedge \langle r \rangle (p \wedge [r]j)$. En este trabajo consideraremos la lógica híbrida básica $\mathcal{H}(@)$, que extiende a la lógica modal básica con nominales y el operador de satisfacción @, el cual permite *evaluar* una fórmula en un elemento particular del modelo.

Formalmente, el conjunto de fórmulas de la lógica $\mathcal{H}(@)$ se define en relación a una signatura $\langle \mathsf{PROP}, \mathsf{NOM}, \mathsf{REL} \rangle$, donde PROP, NOM, REL son conjuntos no vacíos, infinitos numerables y disjuntos dos a dos. PROP es el conjunto de símbolos de proposición, NOM el conjunto de nominales y REL define el conjunto de modalidades. $\mathsf{ATOM} = \mathsf{PROP} \cup \mathsf{NOM}$ es el conjunto de símbolos atómicos. Dada una signatura $\mathcal{S} = \langle \mathsf{PROP}, \mathsf{NOM}, \mathsf{REL} \rangle$ el conjunto de $\mathcal{H}(@)$-fórmulas sobre $\mathcal{S}$ se define inductivamente como $\mathcal{H}(@) = a \mid \neg\varphi \mid \varphi \wedge \varphi' \mid @_n \varphi \mid \langle r \rangle \varphi$, donde $a \in \mathsf{ATOM}$, $n \in \mathsf{NOM}$, $r \in \mathsf{REL}$ y $\varphi, \varphi' \in \mathcal{H}(@)$. Los demás operadores ($\vee$, $\rightarrow$, [ ], etc.) se definen en la forma usual; en particular $[r]\varphi = \neg\langle r \rangle \neg\varphi$.

Definición 1.1. Un *modelo híbrido* es una estructura $M = \langle W, \{R_i\}_{r_i \in \mathsf{REL}}, V \rangle$ donde

$$\begin{aligned} W &\quad \text{es un conjunto no vacío} \\ R_i \subseteq W \times W &\quad \text{es una relación binaria para cada } r_i \in \mathsf{REL} \\ V(p_i) \subseteq W &\quad \text{para cada } p_i \in \mathsf{PROP} \\ V(n_i) = \{w_i\} \subseteq W &\quad \text{para cada } n_i \in \mathsf{NOM} \end{aligned}$$

Definición 1.2. Dado un modelo híbrido $M = \langle W, \{R_i\}_{r_i \in \mathsf{REL}}, V \rangle$, la relación de satisfacibilidad $M, w \models \varphi$ (con $w \in W$) se lee "el modelo M satisface la fórmula φ en w" y se define inductivamente de la siguiente forma:

$$\begin{aligned} M, w \models a_i &\quad \text{sii} \quad w \in V(a_i), a_i \in \mathsf{ATOM} \\ M, w \models \neg\varphi &\quad \text{sii} \quad M, w \not\models \varphi \\ M, w \models \varphi_1 \wedge \varphi_2 &\quad \text{sii} \quad M, w \models \varphi_1 \text{ y } M, w \models \varphi_2 \\ M, w \models \langle r_i \rangle \varphi &\quad \text{sii} \quad \text{existe } w' \in W \text{ tal que } R_i(w, w') \text{ y } M, w' \models \varphi \\ M, w \models @_n \varphi &\quad \text{sii} \quad M, w' \models \varphi, \text{ con } w' \in V(n). \end{aligned}$$

De lo definición anterior, se desprende trivialmente que

$$M, w \models [r_i]\varphi \quad \text{sii} \quad \text{para todo } w' \in W \text{ tal que } R_i(w, w') \text{ y } M, w' \models \varphi$$

La lógica $\mathcal{H}(@)$, a través de fórmulas de la forma $@_i\, j$, donde i y j son nominales, introduce una noción débil de igualdad[1] que no tiene la lógica modal básica y es, por lo tanto, más expresiva que ésta. Sin embargo, es sabido que su problema de satisfacibilidad sigue siendo PSPACE-completo [1].

1.2. El cálculo de resolución para $\mathcal{H}(@)$

Las implementaciones más exitosas de demostradores de teoremas para lógicas modales están basados en el método de tableaux. Los buenos resultados obtenidos hasta el momento se apoyan en el uso de un número importante de heurísticas y refinamientos. Sin embargo, muchas de estas heurísticas dejan de funcionar cuando la lógica subyacente incluye alguna forma de igualdad. Al introducir nominales, la performance de los demostradores basados en tableaux baja considerablemente.

En este escenario, tiene sentido investigar qué sucede con otras familias de algoritmos. En particular, es interesante analizar el *método de resolución* que ha dado los mejores resultados en demostradores de teoremas para la lógica de primer orden con igualdad.

En primer término repasaremos el método de resolución para la lógica de primer orden; a continuación presentamos un cálculo de resolución para $\mathcal{H}(@)$ propuesto en [2]. Finalmente mostramos las limitaciones de dicho cálculo, que son la motivación de esta tesis.

1.2.1. Resolución para lógica de primer orden

El *cálculo de resolución* para lógica de primer orden fue propuesto por Robinson [18] a mediados de la década del '60 y es hoy la base de la gran mayoría de los demostradores automáticos para esta lógica. Para presentar el cálculo, utilizaremos la terminología estándar:

Definición 1.3. Los *términos de primer orden* se definen en forma inductiva: las variables son términos, y si $t_1, \ldots, t_n$ son términos y f es un símbolo de función de aridad n, entonces $f(t_1, \ldots, t_n)$ es un término. Dado un símbolo de relación R de aridad n, y $t_1, \ldots, t_n$, términos de primer orden, se dice que $R(t_1, \ldots, t_n)$ es un *átomo*. Finalmente si X es un átomo, se dice que tanto X como $\neg X$ son *literales*. Una *cláusula de primer orden* es un conjunto de literales; las cláusulas se interpretan como la disyunción de los literales que la conforman.

En resolución se opera sobre un conjunto de cláusulas, que se interpreta como la conjunción de todas las cláusulas. Para obtener esta representación, las fórmulas de entrada del cálculo deben ser pasadas a forma prenexa, *skolemnizadas* (reemplazar cada variable cuantificada existencialmente por un término) y expresadas en forma normal disyuntiva. El cálculo consiste, básicamente, en la saturación del conjunto de cláusulas por medio de una regla de inferencia, y una regla de factorización:

$$\text{(RES)} \quad \frac{C_1 \cup \{\varphi\} \quad C_2 \cup \{\neg\psi\}}{(C_1 \cup C_2)\sigma} \qquad\qquad \text{(FACT)} \quad \frac{C \cup \{\varphi, \psi\}}{(C \cup \{\varphi\})\sigma}$$

donde σ es el unificador más general de los átomos φ y ψ [6]. Ambas reglas se interpretan de la siguiente forma: si se encuentran cláusulas como las de la premisa, se debe agregar una cláusula como la que indica la conclusión al conjunto de cláusulas.

Es fácil convencerse de que la regla de resolución es consistente: si suponemos que $C_1 \cup \{\varphi\}$ y $C_2 \cup \{\neg\psi\}$ son verdaderas, y sabemos que $\varphi\sigma$ y $\neg\psi\sigma$ no pueden ser simultáneamente verdaderas (ya

[1] *Débil* en cuanto siempre debe participar al menos una constante.

que $\varphi\sigma = \psi\sigma$), entonces debemos concluir que o bien C_1 o bien C_2 es verdadera, con lo cual $C_1 \cup C_2$ también debe serlo. La consistencia de la regla de factorización es aún más evidente; ésta es necesaria para garantizar la completitud del cálculo en lógica de primer orden. No es necesaria, por ejemplo, en una versión proposicional del cálculo.

Sea N un conjunto de cláusulas, y sea N^* el conjunto que se obtiene al saturar N con las reglas de resolución y factorización; se puede demostrar que si N no es satisfacible (i.e. no existe un modelo que satisfaga todas las cláusulas), entonces N^* contiene la cláusula vacía [6]. El cálculo de resolución como algoritmo para determinar la satisfacibilidad de una fórmula de la lógica de primer orden φ consiste, entonces, en partir del conjunto inicial de cláusulas asociadas a φ, obtener su saturación con respecto a las reglas de resolución y factorización, y comprobar si este conjunto saturado contiene la cláusula vacía. De hecho, en este algoritmo no es necesario continuar con la saturación del conjunto una vez que se genera la cláusula vacía.

Es interesante notar que mientras que el *peor caso* del método de tableau se da cuando la fórmula de entrada es insatisfacible (ya que es necesario cerrar todas las ramas del árbol), el peor caso del cálculo de resolución se presenta cuando la fórmula de entrada es satisfacible, ya que en este caso debemos obtener la saturación completa del conjunto de cláusulas.

1.2.2. El cálculo de resolución híbrido

En [2] se propone un cálculo basado en resolución que puede ser usado sobre $\mathcal{H}(@)$. La versión que aquí presentaremos del cálculo trabaja con fórmulas en *forma normal negativa*, es decir, sólo es posible aplicar el operador de negación sobre átomos. Esto significa que tanto $\vee$ como $[\cdot]$ serán símbolos *primitivos*.

Definición 1.4. Sea una signatura $\mathcal{S} = \langle \mathsf{PROP}, \mathsf{NOM}, \mathsf{REL} \rangle$, definimos inductivamente el conjunto $\mathcal{H}^{NNF}(@)$ como:

$$\mathcal{H}^{NNF}(@) ::= a \mid \neg a \mid \varphi \vee \varphi' \mid \varphi \wedge \varphi' \mid \langle r \rangle \varphi \mid [r]\varphi \mid @_i\, \varphi$$

donde $a \in \mathsf{PROP} \cup \mathsf{NOM}$, $r \in \mathsf{REL}$, $i \in \mathsf{NOM}$ y $\varphi, \varphi' \in \mathcal{H}^{NNF}(@)$. Además, a las fórmulas de la forma $@_i\, \varphi$ las llamaremos *fórmulas-@*.

Al igual que el cálculo de resolución para lógica de primer orden, el cálculo de resolución híbrido trabaja con conjuntos de *cláusulas*. Una cláusula, en este contexto, es un conjunto de fórmulas-@ arbitrarias pertenecientes a $\mathcal{H}^{NNF}(@)$. Aquí también, una cláusula representa una disyunción, pero no hay ninguna otra restricción en cuanto a la *forma* que deben tener las fórmulas; son las propias reglas del cálculo las que se encargan de armar cláusulas con fórmulas cada vez más simples. Vale notar que considerar sólo fórmulas-@ en las cláusulas no es una restricción en términos de satisfacción: una fórmula φ es satisfacible, si y sólo si para cualquier nominal i que no aparezca en φ, $@_i\, \varphi$ lo es.

Dada una fórmula $\varphi \in \mathcal{H}^{NNF}(@)$, llamamos $\mathit{ClSet}(\varphi) = \{\{@_t\, \varphi\}\}$, donde t es un nominal que no aparece en φ. Podemos ahora definir $\mathit{ClSet}^*(\varphi)$ – el conjunto saturado de cláusulas correspondiente a φ – como el menor conjunto que incluye a $\mathit{ClSet}(\varphi)$ y que está clausurado bajo las reglas de la Figura 1.

Podemos agrupar las reglas del cálculo de acuerdo a la función que cumplen. Las reglas $(\wedge)$, $(\vee)$ y $(@)$ se encargan de simplificar las fórmulas. La regla $(\langle r \rangle)$ es un paso de skolemnización mediante el cual se asigna explícitamente un nombre (un nominal nuevo) a un elemento del modelo previamente cuantificado existencialmente (por medio de un diamante). La regla (RES) es el equivalente de la regla de resolución para lógica de primer orden, mientras que la regla $([r])$ codifica una unificación no trivial y un paso de resolución. Finalmente, las reglas (SYM), (REF) y (PARAM) son el equivalente del conjunto estándar de reglas para manejar igualdad en resolución para lógica de primer orden [5].

La construcción de $\mathit{ClSet}^*(\varphi)$ es un algoritmo correcto y completo para verificar la satisfacibilidad de fórmulas de $\mathcal{H}^{NNF}(@)$: φ es insatisfacible si y sólo si la cláusula vacía $\{\}$ es un elemento

$$(\wedge)\quad \frac{Cl \cup \{@_t\,(\varphi_1 \wedge \varphi_2)\}}{\begin{array}{c} Cl \cup \{@_t\,\varphi_1\} \\ Cl \cup \{@_t\,\varphi_2\} \end{array}} \qquad (\vee)\quad \frac{Cl \cup \{@_t\,(\varphi_1 \vee \varphi_2)\}}{Cl \cup \{@_t\,\varphi_1, @_t\,\varphi_2\}} \qquad (@)\quad \frac{Cl \cup \{@_t\,@_s\,\varphi\}}{Cl \cup \{@_s\,\varphi\}}$$

$$(\text{RES})\quad \frac{Cl_1 \cup \{@_t\,\varphi\} \quad Cl_2 \cup \{@_t\,\neg\varphi\}}{Cl_1 \cup Cl_2}$$

$$([r])\quad \frac{Cl_1 \cup \{@_t\,[r]\varphi\} \quad Cl_2 \cup \{@_t\,\langle r\rangle s\}}{Cl_1 \cup Cl_2 \cup \{@_s\,\varphi\}} \qquad (\langle r\rangle)\quad \frac{Cl \cup \{@_t\,\langle r\rangle\varphi\}}{\begin{array}{c} Cl \cup \{@_t\,\langle r\rangle n\} \\ Cl \cup \{@_n\,\varphi\} \end{array}} \text{ para un } n \text{ nuevo}$$

$$(\text{SYM})\quad \frac{Cl \cup \{@_t\,s\}}{Cl \cup \{@_s\,t\}} \qquad (\text{REF})\quad \frac{Cl \cup \{@_t\,\neg t\}}{Cl} \qquad (\text{PARAM})\quad \frac{Cl_1 \cup \{@_t\,s\} \quad Cl_2 \cup \{\varphi(s)\}}{Cl_1 \cup Cl_2 \cup \{\varphi(s/t)\}}$$

Figura 1: Reglas de resolución para $\mathcal{H}(@)$

de $ClSet^*(\varphi)$. Sin embargo, $ClSet^*(\varphi)$ puede ser un conjunto infinito, por lo tanto, existen fórmulas cuya satisfacibilidad este algoritmo no puede decidir en un número finito de pasos. En la Sección 3 investigamos cómo convertir este cálculo en un método de decisión para $\mathcal{H}(@)$.

1.2.3. Resolución con orden y selección

El cálculo de resolución para lógica de primer orden, tal como fue presentado en la Sección 1.2.1, es refutacionalmente completo; es decir, se puede mostrar que dado un conjunto insatisfacible de cláusulas, el conjunto que se obtiene al saturarlo respecto a la regla de resolución contiene la cláusula vacía [6]. A pesar de ello, no es viable construir demostradores en base a la aplicación directa de estas reglas.

El problema surge de la combinación de dos hechos fácilmente verificables: por un lado, el cálculo permite derivar cláusulas *redundantes*; por el otro, dado un conjunto de cláusulas C, si decimos que $d(C)$ representa el conjunto de todas las cláusulas que se pueden derivar de la aplicación de la regla de resolución exclusivamente sobre cláusulas que aparecen en C, entonces el tamaño de $d(C)$ puede crecer de manera *exponencial* conforme aumenta el número de cláusulas de C. Veamos un ejemplo simple de esto.

Ejemplo 1. Supongamos que N es un conjunto satisfacible de cláusulas formado por:

$$\begin{aligned} C_1 &= \{\neg P(x), \neg S(x)\} \\ C_2 &= \{\neg P(x), S(x)\} \\ C_3 &= \{P(x), Q(x), \neg R(x)\} \\ C_4 &= \{\neg Q(x), S(x)\} \end{aligned}$$

El siguiente es parte del conjunto de cláusulas que el demostrador debería generar a partir de N:

$$\begin{array}{ll} C_5 \equiv \{\neg P(x)\} & (C_2, C_1 \text{usando (RES)}) \\ C_6 \equiv \{Q(x), \neg R(x), \neg S(x)\} & (C_3, C_1 \quad " \quad) \\ C_7 \equiv \{Q(x), \neg R(x), S(x)\} & (C_3, C_2 \quad " \quad) \\ C_8 \equiv \{\neg P(x), \neg Q(x)\} & (C_4, C_1 \quad " \quad) \\ C_9 \equiv \{P(x), \neg R(x), S(x)\} & (C_3, C_4 \quad " \quad) \\ C_{10} \equiv \{Q(x), \neg R(x)\} & (C_5, C_3 \quad " \quad) \\ C_{11} \equiv \{\neg P(x), Q(x), \neg R(x)\} & (C_2, C_6 \quad " \quad) \\ C_{12} \equiv \{\neg R(x), S(x), \neg S(x)\} & (C_6, C_4 \quad " \quad) \\ C_{13} \equiv \{P(x), \neg Q(x), \neg R(x)\} & (C_4, C_6 \quad " \quad) \\ C_{14} \equiv C_{11} & (C_7, C_1 \quad " \quad) \\ C_{15} \equiv C_{12} & (C_7, C_4 \quad " \quad) \end{array}$$

$$
\begin{array}{lll}
C_{16}{\equiv}C_{10} & (C_7, C_6 & \text{"} \quad) \\
C_{17}{\equiv}C_{13} & (C_3, C_8 & \text{"} \quad) \\
C_{18}{\equiv}\{P(x), \neg P(x), \neg R(x)\} & (C_3, C_8 & \text{"} \quad) \\
C_{19}{\equiv}\{\neg P(x), \neg R(x), \neg S(x)\} & (C_6, C_8 & \text{"} \quad) \\
C_{20}{\equiv}\{\neg P(x), \neg R(x), S(x)\} & (C_7, C_8 & \text{"} \quad) \\
\vdots & &
\end{array}
$$

Notemos que C_3 *resuelve* con C_1 eliminando $Q(x)$ para generar C_4 y con C_2 para generar C_5 eliminando $P(x)$. Lo que se debe observar es que esta última cláusula es *redundante*: el objetivo de un demostrador basado en resolución es lograr derivar la cláusula vacía; si suponemos que C_3 puede participar en una derivación de la cláusula vacía, concluiremos que debemos ser capaces de *eliminar* todos los literales que aparecen en C_3. Ahora bien, en C_4 ya habíamos logrado eliminar $Q(x)$, con lo cual convendría intentar eliminar $P(x)$ de esta cláusula como se hace al generar C_7; generar C_5 (ó C_6) resulta redundante. Se puede ver con claridad en este ejemplo cómo las cláusulas redundantes pueden contribuir al rápido crecimiento del conjunto de cláusulas que el demostrador debe manejar.

El mecanismo estándar para controlar la generación de cláusulas en resolución para lógica de primer orden es el llamado *resolución con orden y selección* [6]. La idea general es establecer condiciones bajo las cuáles sea seguro *elegir* un literal de cada cláusula de forma tal que se acepte que la cláusula sea premisa de un paso de inferencia sólo si es para *eliminar* dicho literal.

En el contexto de resolución para lógica de primer orden, decimos que una *función de selección* es una función que asigna a cada cláusula C un conjunto vacío o un conjunto unitario con un literal negativo de C. Dada una función de selección S y cierta relación de orden entre literales $\succ$, el cálculo de resolución para lógica de primer orden con orden y selección se define mediante la regla

$$\text{(RES–OS)} \quad \frac{C_1 \cup \{\varphi\} \quad C_2 \cup \{\neg\psi\}}{(C_1 \cup C_2)\sigma}$$

tal que

1. σ es el unificador más general de los átomos φ y ψ
2. $S(C_1 \cup \{\varphi\}) = \{\}$ y $\varphi\sigma \succ \varphi'$ para todo $\varphi' \in C_1\sigma$
3. $S(C_2 \cup \{\neg\psi\}) = \{\neg\psi\}$ o bien $S(C_2 \cup \{\neg\psi\}) = \{\}$ y $\neg\psi\sigma \succ \psi'$ para todo $\psi' \in C_2\sigma$

Se puede demostrar que el cálculo de resolución con orden y selección (que incluye la regla de factorización) es refutacionalmente completo para la lógica de primer orden cuando se usa un orden $\succ$ con las propiedades adecuadas [6].

Ejemplo 2. A modo de ejemplo, veamos en cuántos pasos llegamos a un conjunto saturado partiendo del mismo conjunto inicial del Ejemplo 1, suponiendo $\neg S(x) \succ S(x) \succ \neg R(x) \succ R(x) \succ \neg Q(x) \succ Q(x) \succ \neg P(x) \succ P(x)$ y una función de selección tal que $S(C) = \{\}$ para todo C. En todos los casos, la fórmula maximal de la cláusula es la que aparece más a la izquierda.

$$
\begin{array}{ll}
C'_5{=}\{\neg P(x)\} & (C_2, C_1 \text{ usando (RES)}) \\
C'_6{=}\{\neg P(x), \neg Q(x)\} & (C_4, C_1 \text{ usando (RES)})
\end{array}
$$

Ninguna otra cláusula puede generarse y el conjunto está saturado.

El Ejemplo 2 muestra en forma elocuente la importancia de contar con un mecanismo de regulación basado en orden y selección si se desea construir un demostrador basado en algún cálculo de resolución. El cálculo de resolución híbrida presentado en la Figura 1 no escapa a este problema. Sin embargo, hasta el momento no se contaba con un mecanismo apropiado de orden y selección para el mismo.

En la Sección 2 proponemos una forma de incorporar orden y selección al cálculo de resolución híbrida y demostramos que con esta extensión se conserva la completitud refutacional.

2. Resolución híbrida con orden y selección

En esta sección veremos cómo logramos incorporar al cálculo de la Figura 1 estrategias de orden y selección como las presentadas en la Sección 1.2.3. Comenzaremos, por dar un tipo especial de orden para fórmulas de $\mathcal{H}^{NNF}(@)$ que nos permite garantizar que el cálculo de resolución con orden y selección que luego proponemos es refutacionalmente completo. La demostración de este hecho es similar, aunque más compleja, a la estándar para el caso de resolución para lógica de primer de orden [6]. Esta demostración requiere una noción adecuada de *modelo de Herbrand*, que presentaremos, previamente, en la Sección 2.3.

2.1. Relaciones de orden entre fórmulas

Se llama *orden* a cualquier relación transitiva e irreflexiva, y se dice que un orden $\succ$ es *total* cuando para cualquier par de elementos x e y, si $x \neq y$, entonces $x \succ y$ o $y \succ x$. Un orden $\succ$ está *bien fundado* cuando no es posible dar con una cadena infinita $x_1 \succ x_2 \succ x_3 \ldots$. Algunos órdenes entre fórmulas son particularmente importantes:

Definición 2.1. Si con $\varphi[\psi]_p$ indicamos que la fórmula ψ es una subfórmula de φ que aparece en posicion p, entonces podemos decir que un orden $\succ$ entre fórmulas tiene la *propiedad de subfórmulas* si $\varphi[\psi]_p \succ \psi$ cuando $\varphi \neq \psi$, y que es un *orden de reescritura* si $\varphi[\psi_1]_p \succ \varphi[\psi_2]_p$ sii $\psi_1 \succ \psi_2$. Un *orden de reducción* es un orden de reescritura bien fundado; y si además tiene la propiedad de subfórmulas se lo llama *orden de simplificación*.

Un método estándar para construir órdenes con estas propiedades es usando el llamado *lexicographic path ordering* (lpo).

Definición 2.2 (Lexicographic Path Ordering). Dado un álfabeto Σ donde cada símbolo tiene asociada una aridad, y dado un orden $\succ$ entre los elementos de Σ al que llamaremos *precedencia*, se define $\succ_{lpo}$, el *lexicographic path ordering* (lpo) entre términos de Σ^* como sigue.

Sean $\varphi = o_1(\varphi_1 \ldots \varphi_m)$ y $\psi = o_2(\psi_1 \ldots \psi_n)$, $n, m \geq 0$ dos términos de Σ^*; diremos que $\varphi \succ_{lpo} \psi$ si y sólo si:

1. $o_1 \succ o_2$ y $\varphi \succ_{lpo} \psi_i$, para todo i tal que $1 \leq i \leq n$; ó
2. $o_1 = o_2$ y para algún j se cumple, $(\varphi_1 \ldots \varphi_{j-1}) = (\psi_1 \ldots \psi_{j-1})$, $\varphi_j \succ_{lpo} \psi_j$ y $\varphi \succ_{lpo} \psi_k$ para todo k tal que $j \leq k \leq n$; ó
3. $\varphi_j \succeq_{lpo} \psi$ para algún j tal que $1 \leq j \leq m$.

Si la precedencia $\succ$ es bien fundada (total), entonces $\succ_{lpo}$ es un orden de simplificación (total) (ver, por ejemplo, [8]). La siguiente es una forma de construir, dado un orden total y bien fundado entre fórmulas, un orden total y bien fundado entre cláusulas (conjuntos finitos de fórmulas):

Definición 2.3. Dado $\succ$ un orden entre fórmulas, se define $\succ_{Cl}$, su extensión para cláusulas, como la relación que cumple: $C_1 \succ_{Cl} C_2$ si y sólo si $C_1 \not\subseteq C_2$ y para toda φ_2 tal que $\varphi_2 \in C_2$ y $\varphi_2 \notin C_1$ existe $\varphi_1 \in C_1$ tal que $\varphi_1 \notin C_2$ y $\varphi_1 \succ \varphi_2$.

De aquí en más, salvo que se indique lo contrario, usaremos el mismo símbolo para una relación de orden entre fórmulas y la extensión para cláusulas de dicho orden.

2.2. Un orden *admisible* para $\mathcal{H}(@)$

En el contexto de los sistemas de resolución se llama *admisible* a un orden entre fórmulas que garantiza que un cálculo con orden y selección conserva la completitud refutacional. En esta sección presentaremos un tipo de orden sobre $\mathcal{H}(@)$ que nos permite definir un cálculo completo con orden y selección basado en el cálculo híbrido. De aquí en más, salvo que se aclare lo contrario, trabajaremos únicamente con

fórmulas bien formadas de $\mathcal{H}^{NNF}(@)$. En este contexto, dado que definiremos un orden basado en lpo, nos resultará conveniente considerar a @, $\langle\rangle$ y [] como operadores *binarios*: $@(\cdot,\cdot) : \mathcal{H}^{NNF}(@) \times \mathsf{NOM} \to \mathcal{H}^{NNF}(@)$[2], $\langle\rangle(\cdot,\cdot) : \mathsf{REL} \times \mathcal{H}^{NNF}(@) \to \mathcal{H}^{NNF}(@)$, $[](\cdot,\cdot) : \mathsf{REL} \times \mathcal{H}^{NNF}(@) \to \mathcal{H}^{NNF}(@)$, por más que usemos la notación estándar $@_n\,\varphi$, $\langle r\rangle\varphi$ y $[r]\varphi$.

Definición 2.4. Un orden $\succ$ es admisible si cumple simultáneamente con estas condiciones, para todo $\varphi, \psi \in \mathcal{H}^{NNF}(@)$:

- A1) es un orden de simplificación total
- A2) $\varphi \succ i$ para todo $\varphi \notin \mathsf{NOM}$ y todo $i \in \mathsf{NOM}$
- A3) si $\varphi \succ \psi$, entonces $@_i\,\varphi \succ @_j\,\psi$, para todo $i, j \in \mathsf{NOM}$
- A4) si $\langle r\rangle i$ es una subfórmula propia de φ con $i \in \mathsf{NOM}$, entonces $\varphi \succ \langle r\rangle j$ para todo nominal j
- A5) $[r]i \succ \langle r\rangle j$, para todo $i, j \in \mathsf{NOM}$

La condición A2 pide que un nominal sea menor que cualquier fórmula que no sea otro nominal, A3 pide que el operador @ no afecte al orden entre fórmulas, A4 introduce una noción muy débil de *complejidad estructural* y A5 prioriza a [] por sobre $\langle\rangle$.

Es importante observar que si $\succ$ cumple con las condiciones de la Definición 2.4, entonces $i \succ j$ si y sólo si $@_j\,i \succ @_i\,j$ para todo par de nominales i y j. Esto significa que un orden admisible ordena en forma coherente las igualdades.

Debemos mostrar que las condiciones de la Definición 2.4 no son demasiado restrictivas. Para ello construimos un orden concreto que cumple con las condiciones pedidas. Nos basamos en el orden presentado en la Definición 2.2, aplicado sobre el alfabeto $\Sigma = \mathsf{PROP} \cup \mathsf{NOM} \cup \mathsf{REL} \cup \{\neg, \wedge, \vee, @, \langle\rangle, []\}$ (notar que $\mathcal{H}^{NNF}(@) \subset \Sigma^*$).

Definición 2.5. Dada una signatura $\langle \mathsf{PROP}, \mathsf{NOM}, \mathsf{REL}\rangle$ con $\mathsf{PROP} = \{P_i \mid i \in \mathbb{N}\}$, $\mathsf{NOM} = \{N_i \mid i \in \mathbb{N}\}$ y $\mathsf{REL} = \{R_i \mid i \in \mathbb{N}\}$, definimos la relación $> \subseteq \Sigma \times \Sigma$ como la clausura transitiva del conjunto:

$$\begin{array}{ll} \{(@, \neg), (\neg, \wedge), (\wedge, \vee), (\vee, []), ([], \langle\rangle)\} & \bigcup \\ \{(\langle\rangle, R_i), (R_i, P_j), (P_j, N_k) \mid i, j, k \in \mathbb{N}\} & \bigcup \\ \{(R_i, R_j), (P_i, P_j), (N_i, N_j) \mid i > j\} & \end{array}$$

Por definición, $>$ es total, irreflexiva y está bien fundada. Sea $\succ_{lpo}$ el orden lexicográfico sobre Σ^* que usa $>$ como precedencia. De acuerdo a lo que hemos visto, $\succ_{lpo}$ es un orden de simplificación total. Finalmente, definimos $\succ_h$ como el orden que verifica $\varphi \succ_h \psi$ si y sólo si $\mathit{size}(\varphi) > \mathit{size}(\psi)$, ó $\mathit{size}(\varphi) = \mathit{size}(\psi)$ y $\varphi \succ_{lpo} \psi$, donde $\mathit{size}(\varphi)$ es la complejidad (en cuanto a cantidad de operadores) de φ.

Proposición 1. *$\succ_h$ es un orden admisible.*

Es interesante observar que no es posible construir un orden admisible usando *únicamente* lpo. Basta con ver que no se puede encontrar una manera de garantizar $\langle\rangle(r', \langle\rangle(r, i)) \succ_{lpo} \langle\rangle(r, j)$ cuando $r \succ_{lpo} r'$ y $j \succ_{lpo} i$, lo cual viola A4.

En lo que resta, usaremos $\succ$ para referirnos a algún orden admisible salvo que se indique específicamente lo contrario.

2.3. Un Teorema de Herbrand para $\mathcal{H}(@)$

Como ya mencionamos, la demostración estándar de completitud refutacional del cálculo de resolución para lógica de primer orden utiliza modelos de Herbrand. Como parte de la adaptación de esta

[2]El orden de los parámetros de este operador ha sido cuidadosamente elegido para simplificar la Definición 2.5 y la subsecuente demostración de la Proposición 1.

demostración al caso $\mathcal{H}(@)$, fue necesario previamente dar con un resultado análogo al Teorema de Herbrand para esta lógica. En esta sección comenzamos por repasar el Teorema de Herbrand clásico para luego presentar este resultado para el caso $\mathcal{H}(@)$.

Dada una signatura de primer orden (sin igualdad) $\mathcal{S} = \langle \mathsf{FUNC}, \mathsf{REL} \rangle$, un *modelo de Herbrand* [11] para $\mathcal{S}$ se define como la interpretación $I = \langle D, \cdot^I \rangle$ donde D es el conjunto de términos de $\mathcal{S}$, $t^I = t$ para todo término t, y $R^I \subseteq D^n$ para todo símbolo de relación R de aridad n.

Es decir, todos los modelos de Herbrand tienen el mismo dominio e interpretan los términos de la misma manera; la única variación está en la forma en que interpretan los símbolos de REL. Notar que podemos identificar un modelo de Herbrand con el conjunto de literales positivos que son verdaderos en el modelo. La importancia de los modelos de Herbrand está dada por el siguiente teorema:

Teorema 2 (Teorema de Herbrand). *Una teoría de primer orden T sobre la signatura $\mathcal{S} = \langle \mathsf{FUNC}, \mathsf{REL} \rangle$ tiene un modelo si y sólo si tiene un modelo de Herbrand sobre la signatura $\mathcal{S}' = \langle \mathsf{FUNC} \cup \mathsf{FUNC}', \mathsf{REL} \rangle$, donde FUNC' es un conjunto infinito numerable disjunto de FUNC.*

El Teorema 2 asegura que para determinar la satisfacibilidad de una teoría de primer orden, alcanza con considerar solamente los posibles modelos de Herbrand (formulaciones alternativas de este teorema en [7]). Para extender este teorema a la lógica $\mathcal{H}(@)$ empezamos por definir la noción apropiada de literal positivo.

Definición 2.6. Dada una signatura híbrida $\mathcal{S} = \langle \mathsf{PROP}, \mathsf{NOM}, \mathsf{REL} \rangle$, definimos LIT-P, el conjunto de *literales positivos* de $\mathcal{H}(@)$ sobre $\mathcal{S}$, como aquel formado por las fórmulas de la forma $@_i\, j$, $@_i\, p$, y $@_i\, \langle r \rangle j$, donde $i, j \in \mathsf{NOM}$, $p \in \mathsf{PROP}$ y $r \in \mathsf{REL}$.

Definición 2.7. Sea $\mathcal{S} = \langle \mathsf{PROP}, \mathsf{NOM}, \mathsf{REL} \rangle$ una signatura híbrida dada. Un *modelo de Herbrand híbrido* para $\mathcal{H}(@)$ en $\mathcal{S}$ es un conjunto $I \subset \mathsf{LIT\text{-}P}$.

Como anteriormente, identificamos un modelo de Herbrand con un conjunto de literales positivos que el modelo satisface y que define unívocamente un determinado modelo híbrido. Dado I un modelo de Herbrand, sea $\sim_I$ (la *relación de equivalencia inducida por I*) la mínima relación de equivalencia que extiende el conjunto $\{(i,j) \mid @_i\, j \in I\} \cup \{(i,i) \mid i \in \mathsf{NOM}\}$. Definimos ahora el modelo híbrido unívocamente determinado por I como la estructura $\langle W^I, \{r_i^I\}, V^I \rangle$, donde

$$\begin{array}{lcl} W^I & = & \mathsf{NOM}/_{\sim_I} \\ r_i^I & = & \{([j],[k]) \mid @_j\, \langle r_i \rangle k \in I\} \\ V^I(p) & = & \{[j] \mid @_j\, p \in I\}, p \in \mathsf{PROP} \\ V^I(i) & = & \{[i]\}, i \in \mathsf{NOM}. \end{array}$$

donde $\mathsf{NOM}/_{\sim_I}$ es el conjunto de clases de equivalencia definido por $\sim_I$ sobre NOM y $[i]$ es la clase de equivalencia asociada a i por $\sim_I$. De ahora en más no diferenciaremos entre un modelo de Herbrand híbrido I y su modelo asociado; y diremos, por ejemplo, que una fórmula $@_i\varphi$ es válida en I, cuando el modelo asociado la satisfaga (siempre nos referiremos a fórmulas de la forma $@_i\varphi$ para que no sea necesaria la referencia a un punto de evaluación en el modelo).

El modelo asociado a I es más complejo que el que definimos para una signatura de primer orden porque $\mathcal{H}(@)$ contiene igualdad y necesitamos entonces particionar el dominio bajo la relación de identidad definida por I. Podemos ahora enunciar el equivalente al Teorema 2.

Teorema 3. *Dado T, un conjunto de fórmulas-@ en $\mathcal{H}(@)$ en la signatura $\mathcal{S} = \langle \mathsf{PROP}, \mathsf{NOM}, \mathsf{REL} \rangle$, T tiene un modelo híbrido si y sólo si tiene un modelo de Herbrand híbrido sobre la signatura $\mathcal{S}' = \langle \mathsf{PROP}, \mathsf{NOM} \cup \mathsf{NOM}', \mathsf{REL} \rangle$ donde NOM' es disjunto de NOM, infinito y numerable.*

2.4. Resolución para $\mathcal{H}^{NNF}(@)$ con orden y selección

En resolución para lógica de primer orden, una función de selección puede elegir una fórmula de una cláusula, con la condición de que sea un literal negativo. Si bien en $\mathcal{H}^{NNF}(@)$ no contamos con una noción de literal que sea directamente aplicable, utilizaremos el complemento de la noción de literales positivos de la Definición 2.7.

Definición 2.8. Diremos que S es una *función de selección* si y sólo si, para cualquier cláusula C se cumple $S(C) \subseteq C$, $|S(C)| \leq 1$ y $S(C) \cap \mathsf{LIT\text{-}P} = \emptyset$

Lo que esta definición nos dice es que una función de selección elige *a lo sumo* una fórmula de cada cláusula, y que aquello que seleccione no puede ser un literal positivo. La Figura 2 contiene las reglas para el cálculo. En ella se asume que $S(C)$ es una función de selección y que $\succ$ es un orden admisible (ver Definición 2.4). La premisa principal de cada regla es siempre la que aparece más a la derecha.

$$(\wedge)\quad \frac{Cl \cup \{@_t\,(\varphi_1 \wedge \varphi_2)\}}{\begin{array}{c} Cl \cup \{@_t\,\varphi_1\} \\ Cl \cup \{@_t\,\varphi_2\} \end{array}} \qquad (\vee)\quad \frac{Cl \cup \{@_t\,(\varphi_1 \vee \varphi_2)\}}{Cl \cup \{@_t\,\varphi_1, @_t\,\varphi_2\}}$$

$$(\mathrm{RES})\quad \frac{Cl_1 \cup \{@_t\,\varphi\} \quad Cl_2 \cup \{@_t\,\neg\varphi\}}{Cl_1 \cup Cl_2}$$

$$([r])\quad \frac{Cl_1 \cup \{@_t\,\langle r\rangle s\} \quad Cl_2 \cup \{@_t\,[r]\varphi\}}{Cl_1 \cup Cl_2 \cup \{@_s\,\varphi\}} \qquad (\langle r\rangle)\quad \frac{Cl \cup \{@_t\,\langle r\rangle\varphi\}}{\begin{array}{c} Cl \cup \{@_t\,\langle r\rangle n\} \\ Cl \cup \{@_n\,\varphi\} \end{array}} \quad \begin{array}{l} \text{para un } n \text{ nuevo} \\ \text{y } \varphi \notin \mathsf{NOM} \end{array}$$

$$(@)\quad \frac{Cl \cup \{@_t\,@_s\,\varphi\}}{Cl \cup \{@_s\,\varphi\}} \qquad (\mathrm{REF})\quad \frac{Cl \cup \{@_t\,\neg t\}}{Cl}$$

$$(\mathrm{SYM})\quad \frac{Cl \cup \{@_s\,t\}}{Cl \cup \{@_t\,s\}}\ \text{si } t \succ s \qquad (\mathrm{PARAM})\quad \frac{Cl_1 \cup \{@_s\,t\} \quad Cl_2 \cup \{\varphi(s)\}}{Cl_1 \cup Cl_2 \cup \{\varphi(s/t)\}}\ \begin{array}{l} \text{si } s \succ t \text{ y} \\ \varphi(s) \succ @_s\,t \end{array}$$

Restricciones:

- Si $C = C' \cup \{\varphi\}$ es la premisa principal, entonces o bien $S(C) = \{\varphi\}$ o, en caso contrario, $S(C) = \emptyset$ y $\{\varphi\} \succ C'$
- Si $D = D' \cup \{\psi\}$ es la premisa auxiliar, entonces $\{\psi\} \succ D'$ y $S(D) = \emptyset$

(tanto φ como ψ representan la fórmula que participa en la inferencia)

Figura 2: Reglas de resolución para $\mathcal{H}^{NNF}(@)$ con orden y selección

Como se puede ver, la Figura 2 difiere de la Figura 1 sólo en el agregado de algunas restricciones tanto locales (reglas ($\langle r\rangle$), (SYM) y (PARAM)) como generales. De acuerdo a estas últimas restricciones, en cada cláusula existe siempre una única fórmula que puede participar en alguna inferencia. Nos referiremos a ella como la *fórmula distinguida* (para $\succ$ y S) de una cláusula y la notaremos $dist^{\succ}_{S}(C)$.

Definición 2.9. Dados un orden admisible $\succ$ y una función de selección S, definimos $max^{\succ}(C)$ como la fórmula maximal (respecto a $\succ$) de C, y $dist^{\succ}_{S}(C)$ como la función tal que $dist^{\succ}_{S}(C) = \varphi$ si $S(C) = \{\varphi\}$ o si $S(C) = \emptyset$ y $max^{\succ}(C) = \varphi$.

Se demuestra que un orden admisible garantiza que toda cláusula producto de una inferencia es menor que la cláusula principal que se utilizó en la aplicación de la regla.

Proposición 4. *Si C es la premisa principal de una inferencia que tiene a D como uno de sus consecuentes, entonces $C \succ D$.*

La de demostración de completitud refutacional está basada en la que se da en [6] para el caso de resolución para lógica de primer orden. Esencialmente, damos un mecanismo para construir un modelo de Herbrand a partir de un conjunto arbitrario (y potencialmente infinito) de cláusulas, de forma tal que si la menor de las cláusulas no es verdadera en este modelo, entonces el cálculo permita derivar una nueva cláusula, menor que la anterior, y que tampoco sea satisfecha por el modelo.

De aquí podremos concluir que en un conjunto clausurado respecto a la aplicación de las reglas, o bien este mecanismo nos provee un modelo que lo satisface, o bien, usando la Proposición 4, la cláusula vacía pertenece a este conjunto. Por esta razón, a estos modelos los llamamos *modelos tentativos*.

La definición de modelo tentativo que damos a continuación es más compleja que la utilizada en [6] ya que aquella estaba dada para el caso de lógica de primer orden sin igualdad, mientras que en $\mathcal{H}(@)$ tenemos que tomar en cuenta las igualdades de la forma $@_i\, j$ (con i y j nominales).

Definición 2.10. Dada I, una interpretación de Herbrand híbrida, definimos la sustitución de nominales por nominales $\sigma_I = \{i \mapsto j | i \sim_I j \wedge (\forall k)(k \sim_I j \rightarrow k \succeq j)\}$, que sustituye cada nominal por el menor nominal de su clase, el cual se toma como representante. Donde no hay ambigüedades, usaremos σ en lugar de σ_I.

Definición 2.11. Dada una signatura híbrida $\mathcal{S} = \langle \mathsf{PROP}, \mathsf{NOM}, \mathsf{REL} \rangle$, definimos SIMP, el conjunto de *fórmulas simples* de $\mathcal{H}^{NNF}(@)$ sobre $\mathcal{S}$ (i.e. las que no pueden ser simplificadas por las reglas $(\wedge)$, $(\vee)$, $(\langle r \rangle)$ o (SYM)) como aquel formado por las fórmulas de la forma $@_i\, j$ (con $i \succ j$), $@_i\, p$, $@_i \neg a$, $@_i \langle r \rangle j$ y $@_i [r]\varphi$, donde $i, j \in \mathsf{NOM}$, $p \in \mathsf{PROP}$, $a \in \mathsf{NOM} \cup \mathsf{PROP}$, $r \in \mathsf{REL}$ y $\varphi \in \mathcal{H}^{NNF}(@)$.

Supongamos un conjunto fijo de cláusulas N. Las siguientes tres definiciones deben tomarse como una unidad. Se presentan en forma separada por claridad pero, como se verá, son las tres mutuamente recursivas.

Definición 2.12 (I_C). Sea C una cláusula (no necesariamente perteneciente a N), llamaremos I_C a la interpretación de Herbrand híbrida dada por $\bigcup_{C \succ D} \varepsilon_D$

Definición 2.13 (Forma reducida). Sean C una cláusula y φ su fórmula maximal. Si $\varphi \in \mathsf{SIMP}$ y se cumple que o bien $\varphi \in \mathsf{LIT\text{-}P}$ y $\varphi = \varphi\sigma_{I_C}$, o bien $\varphi = @_i [r]\psi$ y $i = i\sigma_{I_C}$, entonces decimos que tanto φ como C se encuentran en *forma reducida*.

Definición 2.14 (ε_C). Sea C una cláusula (no necesariamente de N); si se verifica simultáneamente:

1. $C \in N$
2. C está en forma reducida
3. $max^{\succ}(C) \in \mathsf{LIT\text{-}P}$
4. C es falsa en I_C
5. $S(C) = \emptyset$

entonces $\varepsilon_C = \{\varphi\}$; en caso contrario, ε_C es el conjunto vacío.

Decimos que C *produce* φ si $\varepsilon_C = \{\varphi\}$ y la llamaremos una *cláusula productiva*. I_C es la *interpretación parcial de N por debajo de C*. Sólo las cláusulas cuya fórmula maximal φ sea un literal positivo y no tengan fórmulas seleccionadas pueden ser productivas.

Definición 2.15. I_N, un modelo tentativo de N, se define como $\bigcup_{C \in N} \varepsilon_C$.

Si una cláusula C es falsa en I, decimos que C es un *contraejemplo* de I. Analizando cada regla del cálculo y considerando por separado las fórmulas distinguidas de una cláusula que no se encuentran en forma reducida, se obtiene el siguiente resultado.

Teorema 5. *Sea N un conjunto de cláusulas y sea C el contraejemplo mínimo de I_N, respecto de un orden admisible $\succ$. Si $C \neq \{\}$, entonces existe una inferencia usando alguna de las reglas del cálculo tal que:*

1. *C es la premisa principal*
2. *la premisa auxiliar (en los casos que corresponde) es una cláusula productiva*
3. *todos los consecuentes son menores, respecto de $\succ$, que C y al menos uno es un contraejemplo de I_N.*

Usando este teorema, podemos probar la completitud del cálculo.

Teorema 6. *El cálculo de resolución híbrido con orden y selección es refutacionalmente completo.*

3. Terminación del cálculo

El cálculo de resolución que hemos visto constituye, dado el Teorema 6, un método correcto y completo para resolver el problema de determinar si cierta fórmula de $\mathcal{H}(@)$ es satisfacible. En esta sección mostramos cómo convertir este cálculo en un *método de decisión* para dicho problema. Se dice que un método es de *decisión* para el problema P si para *cualquier* instancia de P podemos obtener una respuesta *correcta* acerca de P en un *número finito de pasos*. La consistencia y completitud refutacional del cálculo garantizan, respectivamente, que la respuesta sea *correcta* para *cualquier* instancia del problema. Lo que nos interesa ver, entonces, es cómo garantizar que ésta se obtenga en un *número finito de pasos*.

Lo que buscamos, concretamente, es lograr que para cualquier fórmula $\varphi \in \mathcal{H}(@)$, $ClSet^*(\varphi)$ sea un conjunto *finito*. Cuando esto se cumple, es fácil implementar un algoritmo que compute en tiempo finito $ClSet^*(\varphi)$ (la forma estándar es usando el "algoritmo de la cláusula dada" [20]).

El cálculo de la Figura 1 claramente puede generar un conjunto saturado de fórmulas que sea infinito. Basta con observar que la regla $(\langle r\rangle)$ se aplica aun sobre fórmulas de la forma $@_i \langle r\rangle j$ donde j es un nominal. El cálculo con orden y selección que presentamos en la Figura 2, en cambio, evita la aplicación de $(\langle r\rangle)$ a fórmulas de este estilo y, sin embargo, como mostramos en la Sección 3.1, también con este cálculo es posible generar un conjunto saturado infinito. Luego proponemos una variación del cálculo de la Figura 2 que preserva completitud con el cual podemos garantizar terminación. En todo momento, asumiremos dado un orden admisible $\succ$ (según la Definición 2.4).

3.1. Cómo generar infinitas fórmulas

Ejemplo 3. Tomemos la fórmula satisfacible $@_i\,([r](i \wedge \langle r\rangle p) \wedge \langle r\rangle p)$ y asumamos que i es el menor nominal de un orden admisible. Veamos qué sucede cuando aplicamos las reglas del cálculo sobre esta fórmula:

$$
\begin{array}{rll}
C = \{@_i\,([r](i \wedge \langle r\rangle p) \wedge \langle r\rangle p)\ \} & & \\
D_{1a} = \{@_i\,[r](i \wedge \langle r\rangle p)\} & (C, \text{ usando } (\wedge)) \\
D_{1b} = \{@_i\,\langle r\rangle p\} & & \\
D_{2a} = \{@_i\,\langle r\rangle k\} & (D_{1b}, \text{ usando } (\langle r\rangle)) \\
D_{2b} = \{@_k\,p\} & & \\
D_3 = \{@_k\,(i \wedge \langle r\rangle p)\} & (D_{1a} \text{ y } D_{2a}, \text{ usando } ([r])) \\
D_{4a} = \{@_k\,i\} & (D_3, \text{ usando } (\wedge)) \\
D_{4b} = \{@_k\,\langle r\rangle p\} & & \\
D_{5a} = \{@_k\,\langle r\rangle k_2\} & (D_{4b}, \text{ usando } (\langle\rangle)) \\
D_{5b} = \{@_{k_2}\,p\} & & \\
D_6 = \{@_i\,\langle r\rangle k_2\} & (D_{4a} \text{ y } D_{5a}, \text{ usando (PARAM)}) \\
\vdots & &
\end{array}
$$

Si comparamos D_{2a} y D_6 veremos que en esta derivación son cláusulas equivalentes. Es decir, podemos

repetir los pasos llevados a cabo para derivar D_6 a partir de D_{1a} y D_{2a} para generar $D_{10} = \{@_i \langle r \rangle k_3\}$, y a partir de D_{10} derivar $D_{14} = \{@_i \langle r \rangle k_4\}$ y así sucesivamente. Claramente, la saturación del conjunto inicial $\{C\}$ constituye un conjunto infinito.

Veamos más en detalle la fórmula inicial del Ejemplo 3. Para empezar, es satisfacible sólo en modelos donde p es verdadero en el elemento asociado a i, y éste esté relacionado con sí mismo y sólo con sí mismo. En segundo lugar, esta fórmula tiene dos niveles de profundidad modal, esto significa, intuitivamente, que está *predicando* sobre elementos que están a no más de dos "pasos de distancia" de i.

Por otro lado, observemos qué sucede con k, k_2, k_3, etc. k_2 es un nominal que se introduce a partir de $@_k \langle r \rangle p$, *bajo la hipótesis de que i y k son distintos*. Con esto nos referimos a que, dado que sobre D_{4b} es posible aplicar paramodulación con D_{4a}, podemos pensar que a partir de este hecho se abren dos ramas: aquella en que aplicamos paramodulación y aquella en que no. En la rama en que lo hacemos, estamos asumiendo que i y k están asociados al mismo elemento del dominio; en la rama en que no, estamos asumiendo que son distintos. Este último es el caso que estamos considerando. Ahora bien, k está a un paso de distancia de i, y si son distintos, k_2 debe estar a dos pasos de distancia de i. Sin embargo, en D_6 al reemplazar k por i estamos *anulando* la hipótesis que dio origen a k_2. Es decir, k_2 es un nominal que se crea con la idea (equivocada) de que esté a dos pasos de distancia de i.

En definitiva, hemos visto un ejemplo en el que se obtiene un conjunto $ClSet^*(\varphi)$ infinito mediante la generación de sucesivos nominales nuevos cada vez más *alejados* de aquellos presentes en φ. El siguiente ejemplo nos muestra que también podemos obtener un número infinito de nominales sin necesidad de que éstos estén cada vez más lejos.

Ejemplo 4. Consideremos la siguiente variación de la fórmula del Ejemplo 3: $@_i([r](i \wedge (q \vee \langle r \rangle p)) \wedge \langle r \rangle p)$. Ésta también es satisfacible y permite generar la siguiente derivación:

$$
\begin{array}{rll}
C = \{@_i([r](i \wedge (q \vee \langle r \rangle p)) \wedge \langle r \rangle p)\} & & \\
D_{1a} = \{@_i [r](i \wedge (q \vee \langle r \rangle p))\} & (C, \text{ usando } (\wedge)) \\
D_{1b} = \{@_i \langle r \rangle p\} & & \\
D_{2a} = \{@_i \langle r \rangle k\} & (D_{1b}, \text{ usando } (\langle r \rangle)) \\
D_{2b} = \{@_k p\} & & \\
D_3 = \{@_k (i \wedge (q \vee \langle r \rangle p))\} & (D_{1a} \text{ y } D_{2a}, \text{ usando } ([r])) \\
D_{4a} = \{@_k i\} & (D_3, \text{ usando } (\wedge)) \\
D_{4b} = \{@_k (q \vee \langle r \rangle p)\} & & \\
D_5 = \{@_k q, @_k \langle r \rangle p\} & (D_{4b}, \text{ usando } (\vee)) \\
D_6 = \{@_k q, @_i \langle r \rangle p\} & (D_{4a} \text{ y } D_5, \text{ usando (PARAM)}) \\
D_{7a} = \{@_k q, @_i \langle r \rangle k_2\} & (D_6, \text{ usando } (\langle r \rangle)) \\
D_{7b} = \{@_k q, @_{k_2} p\} & & \\
D_8 = \{@_k q, @_{k_2} (i \wedge (q \vee \langle r \rangle p))\} & (D_{1a} \text{ y } D_{7a}, \text{ usando } ([r])) \\
D_{9a} = \{@_k q, @_{k_2} i\} & (D_8, \text{ usando } (\wedge)) \\
D_{9b} = \{@_k q, @_{k_2} (q \vee \langle r \rangle p)\} & & \\
D_{10} = \{@_k q, @_{k_2} q, @_{k_2} \langle r \rangle p\} & (D_{9b}, \text{ usando } (\vee)) \\
D_{11} = \{@_k q, @_{k_2} q, @_i \langle r \rangle p\} & (D_{9a} \text{ y } D_{10}, \text{ usando (PARAM)}) \\
D_{12a} = \{@_k q, @_{k_2} q, @_i \langle r \rangle k_3\} & (D_{11}, \text{ usando } (\langle r \rangle)) \\
D_{12b} = \{@_k q, @_{k_2} q, @_{k_3} p\} & & \\
\vdots & &
\end{array}
$$

En este caso las cláusulas a comparar son D_{2a}, D_{7a} y D_{12a}, y conviene mirar cómo estas últimas se generan a partir de D_{1b}, D_6 y D_{11}.

Este ejemplo muestra cómo se pueden obtener infinitas apariciones de $@_i \langle r \rangle p$, y cómo cada una de ellas genera a su vez un nuevo nominal a un paso de distancia respecto de i.

3.2. Restringiendo la generación de nuevos nominales: la regla ($\langle r \rangle$)

En la sección anterior presentamos dos formas distintas en que el cálculo de la Figura 2 puede generar infinitos nominales. En una de ellas, se crean nuevos nominales cada vez más alejados; mientras que en la otra, infinitas apariciones de una misma fórmula generan infinitos nominales equidistantes. En esta sección presentamos una variación del cálculo que evita este último comportamiento.

Consideremos la cláusula $C = \{@_i \langle r \rangle p, @_j (p \wedge q)\}$ y supongámosla parte de un conjunto de cláusulas. Dependiendo de la función de selección que utilicemos, una derivación a partir de C comenzará simplificando $@_i \langle r \rangle p$ ó $@_j (p \wedge q)$. Veamos dos posibles derivaciones, una empezando en cada fórmula. Por un lado, tenemos:

$$\begin{array}{lll} C & = \{@_i \langle r \rangle p, @_j (p \wedge q)\} & \\ C_{1a} & = \{@_i \langle r \rangle k, @_j (p \wedge q)\} & (C, \text{ usando } (\langle r \rangle)) \\ C_{1b} & = \{@_k p, @_j (p \wedge q)\} & \\ C_{2a} & = \{@_i \langle r \rangle k, @_j p\} & (C_{1a}, \text{ usando } (\wedge)) \\ C_{2b} & = \{@_i \langle r \rangle k, @_j q\} & \\ C_{3a} & = \{@_k p, @_j p\} & (C_{1b}, \text{ usando } (\wedge)) \\ C_{3b} & = \{@_k p, @_j q\} & \end{array}$$

mientras que por el otro obtenemos:

$$\begin{array}{lll} C & = \{@_i \langle r \rangle p, @_j (p \wedge q)\} & \\ C'_{1a} & = \{@_i \langle r \rangle p, @_j p\} & (C, \text{ usando } (\wedge)) \\ C'_{1b} & = \{@_i \langle r \rangle p, @_j q\} & \\ C'_{2a} & = \{@_i \langle r \rangle k, @_j p\} & (C'_{1a}, \text{ usando } (\langle r \rangle)) \\ C'_{2b} & = \{@_k p, @_j p\} & \\ C'_{3a} & = \{@_i \langle r \rangle l, @_j q\} & (C'_{1b}, \text{ usando } (\langle r \rangle)) \\ C'_{3b} & = \{@_l p, @_j q\} & \end{array}$$

Si comparamos las últimas cuatro cláusulas de cada caso observaremos que sólo difieren en el hecho de que en la segunda derivación se generó un nominal nuevo (l) mientras que en la primera se usó en todos los casos el mismo nominal (k); i.e., la segunda derivación está considerando modelos más complicados que la primera (e.g. modelos en los que i se relaciona con más de un elemento).

De hecho, en el segundo caso estamos *perdiendo información*: las apariciones de $@_i \langle r \rangle p$ en C'_{1a} y C'_{1b} corresponden a la misma fórmula (puesto que ambas derivan de la aparición de $@_i \langle r \rangle p$ en C) y sin embargo ya no es posible relacionar las fórmulas que se derivan de ellas.

Lo que este ejemplo nos sugiere es que dada cualquier aparición de una fórmula $@_i \langle r \rangle \psi$, alcanza con tener un *único* nominal "testigo", que sea sucesor de i y que satisfaga ψ. Más formalmente, podemos demostrar lo siguiente.

Teorema 7. *Sea φ una fórmula de $\mathcal{H}^{NNF}(@)$ y sea j un nominal que no aparece en φ. φ es satisfacible si y sólo si para todo nominal i, toda relación r y toda fórmula $\psi \in \mathcal{H}^{NNF}(@)$, $\varphi[@_i \langle r \rangle \psi / (@_i \langle r \rangle j \wedge @_j \psi)]$ es satisfacible.*

Con este resultado, podemos proponer una regla alternativa para tratar diamantes. Para ello, dividamos NOM en dos conjuntos (infinitos) disjuntos NOM_i (*nominales iniciales*) y NOM_c (*nominales del cálculo*), y llamemos $\mathcal{H}_0^{NNF}(@)$ al conjunto de fórmulas de $\mathcal{H}^{NNF}(@)$ en el que sólo aparecen nominales de NOM_i. De aquí en más supondremos, sin perder generalidad, que el cálculo se utiliza sólo sobre fórmulas de $\mathcal{H}_0^{NNF}(@)$ y que todo nominal que aparece en $ClSet^*(\varphi)$ pero no en φ pertenece a NOM_c.

Definamos, además, $\mathcal{H}^{NNF}_{@\diamond}(@) = \{@_i \langle r \rangle \psi \mid i \in \mathsf{NOM}, r \in \mathsf{REL}, \psi \in \mathcal{H}_0^{NNF}(@), \psi \notin \mathsf{NOM}\}$, es decir, $\mathcal{H}^{NNF}_{@\diamond}(@)$ es el conjunto de fórmulas-@ de $\mathcal{H}^{NNF}(@)$ que pueden ser premisa de la regla ($\langle r \rangle$). Finalmente, sea $nom(x) : \mathcal{H}^{NNF}_{@\diamond}(@) \to \mathsf{NOM}_c$ una función *inyectiva* cualquiera. Definimos la regla alternativa ($\langle r \rangle'$) como:

$$(\langle r\rangle') \quad \frac{Cl \cup \{@_t \langle r\rangle\varphi\}}{\begin{array}{c} Cl \cup \{@_t \langle r\rangle n\} \\ Cl \cup \{@_n \varphi\} \end{array}} \quad \begin{array}{l} \text{con } n = nom(@_t \langle r\rangle\varphi) \\ \text{y } \varphi \notin \mathsf{NOM} \end{array}$$

El Teorema 7 garantiza la consistencia de esta regla. Además, se comprueba que la misma argumentación utilizada para la regla $(\langle r\rangle)$ en la demostración del Teorema 5 vale para la regla $(\langle r\rangle')$. Esto garantiza que al reemplazar $(\langle r\rangle)$ por esta regla se preserva la completitud refutacional del cálculo.

3.3. Restringiendo la generación de fórmulas repetidas: la regla (PARAM)

La regla $(\langle r\rangle')$ evita una derivación infinita como la del Ejemplo 4; sin embargo el uso de esta regla no modifica sustancialmente la derivación del Ejemplo 3, donde se siguen generando infinitos nominales cada vez más alejados de i.

Miremos más en detalle la cláusula D_6 de dicha derivación, que como ya mencionamos, cumple un rol principal en la derivación *cíclica* que genera un conjunto infinito de cláusulas. Esta cláusula se genera a partir de la paramodulación entre las cláusulas D_{4a} y D_{5a}. La fórmula $@_i \langle r\rangle k_2$, sintetizada en dicha operación, es el resultado de asumir que i y k representan lo mismo. Ahora bien, en $D_{5a} = \{@_k \langle r\rangle k_2\}$, k_2 es un nominal introducido bajo la suposición de que k se relaciona con él, y tal que en él p es verdadero. Aquí es importante observar que $nom(@_k \langle r\rangle p) = k_2$. Cuando sintetizamos D_6, k_2 pasa a cumplir un rol adicional: ahora también representa el elemento relacionado con i y tal que p es verdadero en él. Sin embargo, el cálculo tenía *reservado* otro nominal para cumplir esta función: $nom(@_i \langle r\rangle p) = k$.

En la sección anterior vimos que es posible asignar, usando la función nom, un *rol* preciso a cada nominal que se crea durante el cálculo y que esto permite que el cálculo pueda limitarse a considerar modelos más simples. El análisis que acabamos de ver muestra cómo al hacer paramodulación sobre fórmulas de la forma $@_i \langle r\rangle j$ donde alguno de los nominales i, j pertenece a NOM_c, también estamos considerando modelos potencialmente más complejos de lo necesario.

Una solución para este problema consiste en tratar las igualdades sobre fórmulas de la forma $@_i \langle r\rangle j$ (con i, j nominales) de manera tal de aprovechar lo que sabemos sobre los nominales que son introducidos por la aplicación de la regla $(\langle r\rangle')$. A este fin, incorporamos una regla especial de paramodulación sobre fórmulas como la de la cláusula C_6. En la Figura 3 presentamos el cálculo donde la regla (PARAM) ha sido reemplazada por (PARAM') y (PARAM-REL). La segunda regla es la que aplica paramodulación en forma especial sobre las fórmulas que acabamos de ver; (PARAM') difiere de (PARAM) únicamente en que no se aplica sobre las cláusulas que trata (PARAM-REL). Es importante notar que ya no nos alcanza con pedir que $nom(x)$ sea una función inyectiva; en este caso necesitamos que sea un tipo especial de función a la que llamamos *distribuidor de nominales*:

Definición 3.1. Una función $f : \mathcal{H}^{NNF}_{@_\diamond}(@) \to \mathsf{NOM}_c$ es un *distribuidor de nominales* si y sólo si

1. f es biyectiva,
2. la clausura transitiva de la relación dada por Rij si y sólo si $j = f(@_i \langle r\rangle\varphi)$ para algún $\langle r\rangle\varphi$ es un orden parcial, y
3. $i \succ j$ si y sólo si $nom(@_i \langle r\rangle\varphi) \succ nom(@_j \langle r\rangle\varphi)$

Las primeras dos condiciones son suficientemente generales y simplifican los argumentos acerca de este tipo de funciones. Lo que hacen es garantizar que todos los elementos de NOM_c puedan aparecer en $ClSet^*(\varphi)$: la primera condición garantiza sobreyectividad, mientras que la segunda evita *ciclos* como en $i = nom(@_i \langle r\rangle\psi)$. La tercer condición es necesaria para garantizar la completitud del cálculo.

Es necesario comprobar que los cambios introducidos a las reglas de paramodulación preservan consistencia y completitud. Para probar consistencia, se usa el hecho de que dado un modelo cualquiera

$$(\wedge)\quad \frac{Cl \cup \{@_t\,(\varphi_1 \wedge \varphi_2)\}}{\begin{array}{c} Cl \cup \{@_t\,\varphi_1\} \\ Cl \cup \{@_t\,\varphi_2\}\end{array}} \qquad (\vee)\quad \frac{Cl \cup \{@_t\,(\varphi_1 \vee \varphi_2)\}}{Cl \cup \{@_t\,\varphi_1, @_t\,\varphi_2\}}$$

$$\text{(RES)}\quad \frac{Cl_1 \cup \{@_t\,\varphi\} \quad Cl_2 \cup \{@_t\,\neg\varphi\}}{Cl_1 \cup Cl_2}$$

$$([r])\quad \frac{Cl_1 \cup \{@_t\,\langle r\rangle s\} \quad Cl_2 \cup \{@_t\,[r]\varphi\}}{Cl_1 \cup Cl_2 \cup \{@_s\,\varphi\}} \qquad (\langle r\rangle')\quad \frac{Cl \cup \{@_t\,\langle r\rangle\varphi\}}{\begin{array}{c} Cl \cup \{@_t\,\langle r\rangle n\} \\ Cl \cup \{@_n\,\varphi\}\end{array}}\ \begin{array}{l}\text{con } \varphi \notin \mathsf{NOM} \text{ y} \\ n = nom(@_t\,\langle r\rangle\varphi)\end{array}$$

$$(@)\quad \frac{Cl \cup \{@_t\,@_s\,\varphi\}}{Cl \cup \{@_s\,\varphi\}} \qquad \text{(REF)}\quad \frac{Cl \cup \{@_t\,\neg t\}}{Cl} \qquad \text{(SYM)}\quad \frac{Cl \cup \{@_s\,t\}}{Cl \cup \{@_t\,s\}}\ \text{si } t \succ s$$

$$\text{(PARAM')}\qquad \frac{Cl_1 \cup \{@_s\,t\} \quad Cl_2 \cup \{\varphi(s)\}}{Cl_1 \cup Cl_2 \cup \{\varphi(s/t)\}}$$

si $s \succ t$, $\varphi(s) \succ @_s\,t$, y cuando $\varphi(s) = @_i\,\langle r\rangle j$, entonces $j \in \mathsf{NOM}_i$, o $t \in \mathsf{NOM}_i$ y $j = s$

$$\text{(PARAM-REL)}\qquad \frac{Cl_1 \cup \{@_s\,t\} \quad Cl_2 \cup \{@_s\,\langle r\rangle n\}}{\begin{array}{c} Cl_1 \cup Cl_2 \cup \{@_t\,\langle r\rangle k\} \\ Cl_1 \cup Cl_2 \cup \{@_n\,k\}\end{array}}$$

si $s \succ t$ y $n \in \mathsf{NOM}_c$, y donde para algún φ, $n = nom(@_s\,\varphi)$, y $k = nom(@_t\,\varphi)$

Restricciones:

- Si $C = C' \cup \{\varphi\}$ es la premisa principal, entonces o bien $S(C) = \{\varphi\}$ o, en caso contrario, $S(C) = \emptyset$ y $\{\varphi\} \succ C'$
- Si $D = D' \cup \{\psi\}$ es la premisa auxiliar, entonces $\{\psi\} \succ D'$ y $S(D) = \emptyset$
- $nom(x)$ es un *distribuidor de nominales*

(tanto φ como ψ representan la fórmula que participa en la inferencia)

Figura 3: Reglas de resolución para $\mathcal{H}(@)$ con orden, selección y control de nominales

de φ y un distribuidor de nominales $nom(x)$, se puede construir otro modelo de φ pero que además cumpla cierto criterio de compatibilidad con $nom(x)$. Esencialmente, si M es *compatible* con $nom(x)$, debe suceder que $M \models @_s\,\langle r\rangle\psi$ y $i = nom(@_s\,\langle r\rangle\psi)$ implique $M \models @_s\,\langle r\rangle i$ y $M \models @_i\,\psi$; y que si, además, $j = nom(@_t\,\langle r\rangle\psi)$ y $M \models @_s\,r$, entonces $M \models @_i\,j$.

Teorema 8. *Si $\varphi \in \mathcal{H}_0^{NNF}(@)$ es satisfacible,* $\mathrm{ClSet}^*(\varphi)$ *(saturado con las reglas de la Figura 3) también lo es. Por lo tanto, el cálculo de la Figura 3 es consistente.*

Lo primero que conviene observar para probar la completitud del nuevo cálculo es que la restricción sobre el orden de los nominales en la Definición 3.1 permite garantizar que los consecuentes que se obtienen al aplicar la regla (PARAM-REL) son menores que su premisa principal. Con esto, podemos adaptar la demostración del Teorema 5, a las reglas nuevas.

Teorema 9. *El cálculo de la Figura 3 es refutacionalmente completo.*

Como parte del trabajo de tesis también demostramos que la Definición 3.1 no es demasiado estricta, es decir, que puede ser satisfecha por alguna función concreta. Para ello, mostramos que para cualquier lenguaje híbrido podemos extender el conjunto de nominales de forma adecuada, de manera que la construcción de un distribuidor de nominales sea trivial.

3.4. Terminación

En esta sección veremos que si usamos un refinamiento de un orden admisible, el número de fórmulas distintas que el cálculo de la Figura 3 puede generar es finito. Con finitas fórmulas sólo es posible armar un conjunto finito de cláusulas distintas; con lo cual, el conjunto saturación debe ser necesariamente finito. Comencemos, entonces, con una observación simple.

Proposición 10. *Sea φ una fórmula de $\mathcal{H}_0^{NNF}(@)$; para toda fórmula $\psi \in \mathrm{ClSet}^*(\varphi)$ se cumple $size(\varphi) \geq size(\psi)$.*

Al igual que en la Definición 2.5, $size(\varphi)$ representa la cantidad de operadores (incluyendo los de aridad cero) de φ. Lo que esta proposición nos dice es que si $ClSet^*(\varphi)$ es infinito, esto no puede deberse a la presencia de fórmulas infinitamente grandes: es necesario que haya infinitos nominales distintos. Lo que veremos a continuación es que los dos ejemplos presentados en la Sección 3.1 ilustran todas las maneras en que pueden generarse infinitos nominales y que los mecanismos propuestos para remediar esto efectivamente funcionan. Para ello nos valemos de una medida de *distancia* entre un nominal cualquiera y algún nominal de NOM_i.

Definición 3.2. Dado un distribuidor de nominales $nom(x)$, la función $level(\cdot) : \mathsf{NOM} \to \mathbb{N}$ se define como:

$$level(i) = \begin{cases} 0 & \text{si } i \in \mathsf{NOM}_i \\ level(j) + 1 & \text{si } i = nom(@_j \langle r \rangle \varphi) \end{cases}$$

Claramente, $n \in \mathsf{NOM}_i$ si y sólo si $level(n) = 0$. Ahora podemos caracterizar fácilmente las dos maneras de generar infinitos nominales. La primera derivación se caracteriza por el hecho de que $level(n)$, cuando n es un nominal que aparece en $ClSet^*(\varphi)$, no está acotada; la segunda, por el hecho de que existe un conjunto infinito N de nominales que aparecen en $ClSet^*(\varphi)$ para el cual se cumple $level(n) = c$ si $n \in N$, para algún $c > 0$. Es claro que si no se cumple ninguna de estas dos condiciones, el conjunto de nominales no puede ser infinito.

Como ya dijimos, la función *level* nos da una medida de *profundidad* de nominales. Esta idea juega un papel importante, y de hecho influye en la noción de orden con la que trabajaremos.

Definición 3.3. Un orden $\succ$ y un distribuidor de nominales $nom(x)$ son *admisibles para terminación* si $\succ$ es un orden *admisible* y además verifica que para todo par de nominales i y j $level(i) > level(j)$ implica $i \succ j$, donde $level(x)$ está definido en función de $nom(x)$.

De aquí en más, asumimos que $\succ$ y $nom(x)$ son, respectivamente, un orden y un distribuidor de nominales de este tipo.

La prueba de terminación consiste en garantizar que, con el cálculo de la Figura 3, $level(n)$ está acotada cuando n aparece en $ClSet^*(\varphi)$ y que en cada *nivel* hay un número finito de nominales. Para ver la primera condición, utilizamos una noción de *profundidad modal* para fórmulas-@ que combina la profundidad meramente sintáctica (i.e., el concepto habitual de "profundidad modal") con la *profundidad* (según la función *level*) del prefijo de la fórmula-@. Para la segunda, nos valemos de una función que nos devuelve todos los nominales de un conjunto de cláusulas que pertenezcan a determinado nivel

Definición 3.4. Para toda fórmula $@_i \varphi$ definimos $d'(@_i \varphi) = level(i) + d(\varphi)$, donde $d(\varphi)$ es la profundidad modal de φ. Además, dado cualquier conjunto de cláusulas N, definimos $level_x(N) = \{i \mid i \in \mathsf{NOM} \wedge level(i) = x \wedge i \in S_f(N)\}$, donde $S_f(N)$ es el conjunto de todas las subfórmulas que aparecen en N.

Teorema 11. *Para toda fórmula φ, el conjunto de nominales distintos que aparecen en $\mathrm{ClSet}^*(\varphi)$ es finito. Por lo tanto, el cálculo de resolución de la Figura 3 constituye un método de decisión para la lógica $\mathcal{H}(@)$.*

La demostración de este teorema se sigue directamente de estas propiedades, que fueron demostradas. En lo que sigue, φ es una fórmula arbitraria de $\mathcal{H}_0^{NNF}(@)$:

1. Para toda cláusula $\{@_i \langle r \rangle j\} \cup C \in \mathit{ClSet}^*(\varphi)$, o bien $\mathit{level}(j) = 0$ o bien $\mathit{level}(j) = \mathit{level}(i)+1$.
2. Para toda fórmula ψ que aparezca en $\mathit{ClSet}^*(\varphi)$, si en ψ aparece algún $i \in \mathsf{NOM}_c$ (i.e. $\mathit{level}(i) > 0$), entonces ψ es de la forma: $@_i \psi'$, $@_n \langle r \rangle i$ ó $@_n i$ ($i \notin S_f(\psi')$ y $i \neq n$).
3. Si ψ aparece en alguna cláusula de $\mathit{ClSet}^*(\varphi)$, entonces $d'(\psi) \leq d(\varphi)$ y, además, si $\psi = @_i j$, entonces $\mathit{level}(j) \leq d(\varphi)$. Luego, para toda fórmula ψ, si i es un nominal que aparece en alguna fórmula de $\mathit{ClSet}^*(\psi)$, entonces $\mathit{level}(i) \leq d(\psi)$.
4. Para todo nominal i, el número de fórmulas distintas de la forma $@_i \langle r \rangle \psi$ (con $\psi \notin \mathsf{NOM}$) que aparecen en $\mathit{ClSet}^*(\varphi)$ es finito.
5. Para todo $x \in \mathbb{N}$, $\mathit{level}_x(\mathit{ClSet}^*(\varphi))$ es un conjunto finito.

4. Implementación y evaluación

HyLoRes es un demostrador para la lógica $\mathcal{H}(@, \downarrow)$ escrito en Haskell, de aproximadamente 5000 líneas de código, basado en el cálculo de resolución presentado en [2]. Es necesario aclarar que no se trata de una herramienta que pretenda competir seriamente con demostradores que representan el estado del arte en demostración automática para lógica de primer orden o DLs. Demostradores como RACER [10, 17] ó *SAT [9, 19] están especialmente afinados y cuentan con una batería de heurísticas y optimizaciones con las que consiguen resultados sobresalientes. En contraste, HyLoRes implementa un conjunto relativamente simple de optimizaciones y es todavía más una prueba de concepto que una aplicación que pueda ser usada en un ambiente real. Pero es justamente por ello que se trata de una entorno ideal para poner a prueba nuevas ideas y realizar una valoración empírica de las mismas.

Como parte de este trabajo se desarrolló una versión de HyLoRes que utiliza las reglas presentadas en la Figura 3 y se realizaron pruebas para comparar la performance de ambas versiones. En esta sección solamente presentamos algunos resultados de las pruebas que realizamos.

Hoy en día, el test estándar para la lógica modal básica es el denominado 3CNF$\Box_m$ aleatorio [16], que es una adaptación del test 3CNF aleatorio de la lógica proposicional [15]. Este tipo de tests se caracterizan por generar lotes de fórmulas aleatorias que se ajusten a determinadas restricciones (e.g., número de variables proposicionales, profundidad modal, número máximo de cláusulas). En general, cuanto mayor es el número de cláusulas de una de estas fórmulas, mayor es la probabilidad de que sea insatisfacible.

La definición estándar de CNF$\Box_m$ aleatorio genera fórmulas estrictamente modales. En [3] se presenta una extensión para trabajar con demostradores para lógicas híbridas. Esta extensión, hCNF$\Box_m$, genera formulas para cualquier sublenguaje de $\mathcal{H}(@, A, \downarrow)$. hGen, una herramienta que genera casos de test a partir de hCNF$\Box_m$, fue utilizada para realizar las pruebas.

Las variables que utilizamos en la generación de lotes de pruebas fueron: "cantidad de variables proposicionales (V)", "cantidad de nominales (N)", "máxima profundidad modal (D)" y "cantidad de cláusulas (L)". Fijados los valores de V, N y D, se generaron, para cada valor de L, lotes de 100 fórmulas, las cuales fueron dadas como entrada a los demostradores, con un timeout de 40 segundos por fórmula. En los gráficos mostramos, por cada uno, el porcentaje de fórmulas que se decidieron satisfacibles, el porcentaje de insatisfacibles y el de timeouts; también se muestra la mediana del tiempo de ejecución.

En la primera prueba, comparamos la performance de ambas versiones del demostrador utilizando fórmulas simples ($V = 2$, $N = 3$, $D = 1$). En este caso la performance de HyLoRes 2.0 fue claramente superior a la de la versión 1.0. En la Figura 4 se ve claramente cómo a HyLoRes 1.0 le cuesta resolver una fracción importante de los problemas más simples, mientras que HyLoRes 2.0 los resuelve todos. Es

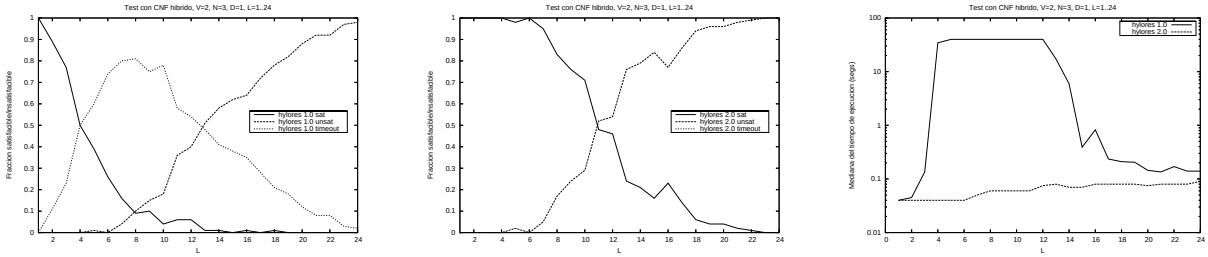


Figura 4: Evaluación con hCNF$\Box_m$ de HyLoRes 1.0 y 2.0 – Fórmulas simples.

interesante notar que HyLoRes 1.0 tiene la mayor proporción de *timeouts* en la región en que la mayoría de las fórmulas son satisfacibles. Se debe tener en cuenta que las restricciones de orden y selección aceleran el proceso de saturación, y que en la versión 1.0 no se contaba con ningún mecanismo de este tipo para la regla de paramodulación.

Como muestra la Figura 4, tener un número importante de *timeouts* afecta negativamente la representatividad de los valores graficados. La curva cortada en el gráfico de cláusulas generadas, y la forma poco usual de la curva del gráfico de tiempo de ejecución son una muestra elocuente de esto. Por otro lado, se puede ver que para valores chicos de L el tiempo de inicialización de HyLoRes 2.0 es mayor al tiempo que necesita para resolver el problema en sí.

Figura 5: Evaluación con hCNF$\Box_m$ de HyLoRes 2.0 – Fórmulas complejas, baja cantidad de nominales.

El número de casos que HyLoRes 1.0 no puede resolver dentro de un tiempo razonable cuando se aumenta la profundidad modal de las fórmulas generadas es demasiado grande como para que podamos tenerlo en cuenta. Es por ello que las últimas pruebas se realizaron sólo sobre distintas configuraciones de la versión 2.0.

En la Figura 5 se muestran los resultados para fórmulas en las que se aumenta la complejidad estrictamente modal: $V = 8$, $N = 3$ y $D = 7$. En este caso, el número de timeouts en la zona más difícil es de un 15 %, sin embargo, el tiempo medio de respuestas sigue sin superar los 2 segundos.

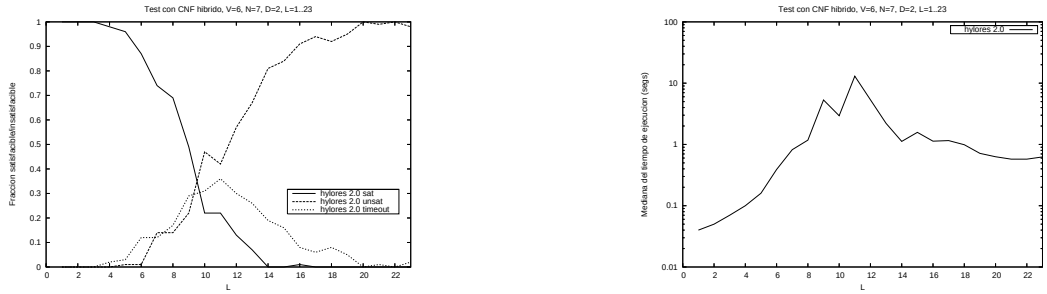


Figura 6: Evaluación con hCNF$\Box_m$ de HyLoRes 2.0 – Fórmulas medianas, alta cantidad de nominales.

Finalmente, en la Figura 6 se muestran los resultados obtenidos con fórmulas en las que aumenta el número de nominales manteniendo baja la profundidad modal: $V = 6$, $N = 7$, $D = 2$. Un número mayor de nominales implica una aplicación más frecuente de las costosas reglas de paramodulación. Como resultado de esto, HyLoRes 2.0 tiene un porcentaje más alto de timeouts con estas fórmulas, y el tiempo medio de ejecución en la zona más difícil supera los 10 segundos. Es importante notar que las fórmulas de la Figura 6 tienen el triple de variables proposicionales y más del doble de nominales que las de la Figura 4, que ya eran demasiado difíciles para HyLoRes 1.0.

Referencias

[1] C. Areces, P. Blackburn, and M. Marx. A road-map on complexity for hybrid logics. In *Proceedings of the 8th Annual Conference of the EACSL*, pages 307–321, 1999.

[2] C. Areces, H. de Nivelle, and M. de Rijke. Resolution in modal, description and hybrid logic. *Journal of Logic and Computation*, 11(5):717–736, 2001.

[3] C. Areces and J. Heguiabehere. hGen: A random CNF formula generator for Hybrid Languages. In *Proceedings of Methods for Modalities 3*, Nancy, France, 2003.

[4] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[5] L. Bachmair and H. Ganzinger. Equational reasoning in saturation-based theorem proving. In *Automated deduction—a basis for applications, Vol. I*, pages 353–397. Kluwer, Dordrecht, 1998.

[6] L. Bachmair and H. Ganzinger. Resolution theorem proving. In J. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 1, chapter 2, pages 19–99. Elsevier, 2001.

[7] S. Buss. On Herbrand's theorem. In *Logic and Computational Complexity*, number 960 in Lecture Notes in Computer Science, pages 195–209. Springer-Verlag, 1995.

[8] N. Dershowitz and J. Jouannaud. Rewrite systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science. Volume B: Formal Models and Sematics (B)*, pages 243–320. Elsevier and MIT Press, 1990.

[9] E. Giunchiglia, A. Tacchella, and F. Giunchiglia. SAT-based decision procedures for classical modal logics. *Journal of Automated Reasoning*, 28(2):143–171, 2002.

[10] V. Haarslev and R. Möller. RACER system description. In *IJCAR 2001*, number 2083 in LNAI, Siena, Italy, June 2001.

[11] J. Herbrand. *Recherches sur la théorie de la démonstrations*. PhD thesis, Sorbone, Paris, 1930. Reprinted in W. Goldfarb, editor, *Logical Writings*. Reidel, 1971.

[12] S. Kripke. A completeness theorem in modal logic. *Journal of Symbolic Logic*, 24:1–14, 1959.

[13] S. Kripke. Semantic analysis of modal logics I, normal propositional calculi. *Zeitschrift fürr mathemathische Logik und Grundlagen der Mathematik*, 9:67–96, 1963.

[14] S. Kripke. Semantical consideration on modal logics. *Acta Philosophica Fennica*, 16:83–94, 1963.

[15] D. Mitchell, B. Sleman, and H. Levesque. Hard and easy distributions of SAT problems. In *Proc. of the 10th National Conference on Artificial Intelligence*, pages 459–465, 1992.

[16] P. Patel-Schneider and R. Sebastiani. A new general method to generate random modal formulae for testing decision procedures. *Journal of Artificial Intelligence Research*, 18:351–389, May 2003.

[17] RACER System Description. `http://www.sts.tu-harburg.de/~r.f.moeller/racer/`, 2004.

[18] J. Robinson. A machine-oriented logic based on the resolution principle. *J. ACM*, 12(1):23–41, Jan. 1965.

[19] *SAT Project. `http://www.mrg.dist.unige.it/~tac/StarSAT.html`, 2004.

[20] A. Voronkov. Algorithms, datastructures, and other issues in efficient automated deduction. In *IJCAR 2001*, number 2083 in LNAI, pages 13–28, Siena, Italy, June 2001.

A MINIMUM INTERFERENCE ROUTING ALGORITHM[1]

Gustavo Bittencourt Figueiredo

gustavo@ic.unicamp.br

Instituto de Computação, Universidade Estadual de Campinas (UNICAMP)

Avenida Albert Einstein, 1251 - Caixa Postal 13084-971 Campinas-SP, Brasil

Tel: +55 (19) 3788-5878 FAX: +55 (19) 3788-5847

Advisor: Nelson L. S. da Fonseca

nfonseca@ic.unicamp.br

Instituto de Computação, Universidade Estadual de Campinas (UNICAMP)

Avenida Albert Einstein, 1251 - Caixa Postal 13084-971 Campinas-SP, Brasil

Tel: +55 (19) 3788-5878 FAX: +55 (19) 3788-5847

Co-Advisor: José A. S. Monteiro

suruagy@unifacs.br

Universidade Salvador (UNIFACS)

R. Ponciano de Oliveira, 126 - Rio Vermelho CEP 41.950-275 Salvador-BA, Brasil

TEL +55 (071) 330-4663 FAX: +55 (071) 330-4666

[1]This work was partially sponsored by the Brazilian National Council for Research (CNPq).

A Minimum Interference Routing Algorithm

Gustavo B. Figueiredo Nelson L. Saldanha da Fonseca

Institute of Computing

State University of Campinas

Brazil

Email: {gustavo, nfonseca}@ic.unicamp.br

José A. Suruagy Monteiro

NUPERC

Salvador University

Brazil

Email: suruagy@unifacs.br

Abstract

The problems of LSP routing and capacity provisioning in MPLS networks were investigated in [1]. A new minimum interference routing algorithm as well as a new mechanism of dynamic sizing of LSPs in MPLS networks were proposed. Howerver, due to space limitations, only the routing algorithm will be presented in this work. Informations regarding dynamic sizing of LSs can be found in [1], [2].

Minimum Interference Routing algorithms aim at reducing rejections of future requests for the establishment of Label Switched Paths (LSPs) but make no assumption about specific patterns of arrival request. This paper introduces a novel minimum interference routing algorithm, Light Minimum Interference Routing (LMIR), which is based on a new approach to the identification of critical links. This approach reduces the computational complexity involved in finding a path for the establishment of an LSP. The LMIR is shown to have the same precision as existing algorithms but with less computational complexity.

I. Introduction

In spite of the increase of Internet link capacity, congestion is still common. One of the main reasons for this is an unbalanced use of resources and Traffi c Engineering if often employed to balance this use by mapping flows into network links. MultiProtocol Label Switching (MPLS) is the key to Traffi c Engineering, since it promotes path establishment, thus guaranteeing the Quality of Service (QoS) required by networks flows.

The use of traditional internet routing algorithms based on the shortest path in MPLS networks leads to a rapid saturation of network links. As a consequence, new alternative algorithms, among them the minimum interference routing algorithms, have been proposed to promote a more balanced utilization of resources. The criteria for selecting a path adopted by these algorithms, consider those paths which will result in the least impact on the rejection of future

requests. However, such criteria, with multiple independent QoS requirements, lead to NP-complete problems [3], [4]. As a consequence, various heuristic algorithms have been proposed to deal with such criteria.

Most of minimum interference algorithms are based on the use of a maximum flow type of algorithm to identify critical links, since these links, which belong to the minimum cut set of this pair of nodes, are what is responsible for a reduction in the maximum flow between a pair of nodes. In such algorithms, a max-flow algorithm is typically executed for all ingress/egress node pairs of a network domain for each LSP establishment request, although this increases the computational complexity of these algorithms.

This paper introduces a new heuristic, the *Light Minimum Interference Routing* (LMIR) algorithm [1], [5], which guarantees minimum interference without the need for the execution of repeated max-flow algorithms. Instead, this algorithm uses a modified version of Dijkstra's algorithm, which is less computationally complex than max-flow algorithms, yet produces similar results. The LMIR algorithm is also independent of the pattern of arrival of LSP establishment requests.

The rest of this paper is structured as follows. Section II introduces the definition of the minimum interference routing problem and describes existing algorithms. Section III introduces the Light Minimum Interference Routing (LMIR) algorithm. Section IV evaluates the performance of the algorithm proposed via simulations. Finally, Section V presents the conclusions.

II. Minimum Interference Routing Algorithms

The central idea behind existing minimum interference routing algorithms is the choice of a path for the establishment of an LSP which minimizes the reduction in maximum flow of other source-destination (SD) pairs. The aim is to increase the number of available paths which can accommodate future LSP establishment requests.

Prior to the presentation of the minimum interference problem itself, certain relationships are defined. These are determined as follows: Let the graph $G = (V, E)$ represent a network, where V is the set of vertices (or nodes) and E the set of edges (or links). The flow is the amount of data transmitted between two nodes of G with a positive capacity, $c(u, v)$ assigned to each link $l = (u, v)$ to denote the maximum amount of flow that can pass through (u, v). Each link has a certain residual capacity, which is defined as $c_r(u, v) = c(u, v) - f(u, v)$ where $f(u, v)$ is the amount of flow passing through (u, v). The residual graph, $G_f(V, E_f)$, is composed of the set of links, E_f, for which residual capacities are greater than zero.

A subset of the nodes, denoted by P, involves the ingress/egress nodes which are the end points of the LSPs. The network topology needs to be known to determine the route between an source-destination pair and it is assumed that a link state routing protocol is in place to diffuse the network topology, as well as a routing database for the storing of the values of residual link capacities.

Each request arrival is specified by three variables: S, D and bw, where S denotes the source node, D the egress destination node, and bw the bandwidth required to carry the traffic between S and D.

The minimum interference routing problem can be stated as follow: *find a path for the establishment of an LSP from a source node to a destination node such that each link along the path has a residual capacity value of at*

least the requested amount of bandwidth. The chosen path should minimize the reduction of the maximum network flow with the restriction that flow splitting is prohibited.

Two major algorithms are presented in this section: the Minimum Interference Routing Algorithm (MIRA) and *Wang, Su and Chen's* algorithm (WSC).

A. The Minimum Interference Routing Algorithm

The maximum flow of an SD pair is reduced whenever the available bandwidth of an link belonging to the minimum cut set is reduced. The Minimum Interference Routing Algorithm (MIRA) [6] avoids links belonging to the min-cut set of other SD pairs when establishing an LSP between a pair of nodes. MIRA assumes that the network topology and the residual bandwidth of the links are known at the time of an LSP establishment.

MIRA consists of three steps, the first involving the computation of the maximum flow between all SD pairs, and the second involving the identification of critical links and the assignment of weights representing priorities in relation to other SD pairs. These weights are computed according to the following equation:

$$w(u,v) = \sum_{(s,d)|(u,v)\in C_{sd}} \alpha_{sd}, \qquad \forall (u,v) \in E, \tag{1}$$

where α_{sd} is the weight of an (s,d) pair and C_{sd} is its set of critical links. The third step of the algorithm involves the execution of the Dijkstra algorithm using $w(u,v)$ as weights.

Kodialam and Lakshman [6] showed that the number of rejections resulting from the use of MIRA is lower than when such minimum interference criteria are not used. The improvement is due mainly to the avoidance of links which would reduce the maximum flow of other pairs, thus, limiting the rejection of many future requests. Evidence of the impact of a reduction in maximum flow on blocking probability was presented in their seminal paper [6], which shows that their algorithm produces lower blocking probability than do algorithms which do not take interference into account.

B. The Wang, Su and Chen Algorithm

The algorithm proposed by *Wang, Su and Chen* [7] is based on MIRA but adopts a different criterion for the identification of critical links, in the computation of weights.

According to Wang et al. [7], the major drawback of MIRA is that critical links identification focuses on a single SD pair and does not detect the critical nature of an link for a group of pairs. Such a drawback can potentially lead to the denial of various requests, especially in networks with concentrating vertices.

In Figure 1, a graph with concentrating vertices is presented. Suppose that $n+1$ requests arrive to the pairs (S_0, D_0), (S_1, D_1),..., (S_n, D_n). The first request demands an LSP with n bandwidth units, while all the rest require LSPs with just one bandwidth unit.

When the first request arrives, MIRA computes the maximum flow and identifies the critical links for all other source-destination pairs, i.e., (S_1, D_1),..., (S_n, D_n). The critical links after the arrival of this first request and, therefore, the only ones with $w((u,v)) \neq 0$ will be (S_1, C),..., (S_n, C) since none of the other links are in the minimum cut set of any source-destination pair. Thus, the weights for all the links of the four available paths

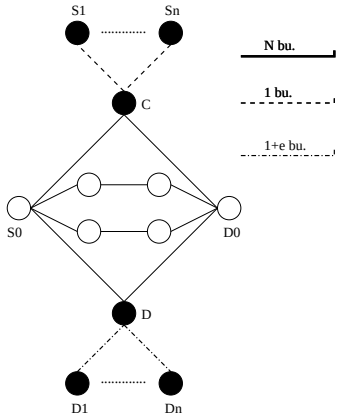


Fig. 1. Graph with concentrating vertices.

connecting S_0 to D_0 will be the same. The chosen path would be the one passing through one of the black vertices (C or D in the figure), since these involve fewest hops. Once an LSP using such path is established, the residual link capacities for this path are updated to 0 and no subsequent request will be accepted.

In order to avoid paths through concentrating vertices, Wang et al., proposed an algorithm with a weight function that takes the link contribution to the maximum flow into consideration. There are two cases to be considered. In the first, the link contribution to the maximum flow is less than its residual capacity which means that the link does not belong to the minimum cut set and will, therefore, receive smaller weighting. In the second case, the link contribution is equal to its max-flow contribution. In this case, the link belongs to the min cut set. Higher weights values are, then, assigned to those links.

In their algorithm, link weights are computed as

$$w(u,v) = \sum_{(s',d') \in P} \frac{f_{(u,v)}^{s'd'}}{\theta^{s'd'} \cdot c_r(u,v)}, \qquad (u,v) \in E, \tag{2}$$

where:

- $c_r(u,v)$ is the residual link capacity,
- $\theta^{s'd'}$ represents the maximum flow between pair (s',d'),
- $f_{(u,v)}^{s'd'}$ is the contribution of the link to the maximum flow of (s',d').

Links with residual capacities smaller than the requested bandwidth are eliminated and the Dijkstra algorithm is executed using weights $(w(u,v))$ as link costs [7]. Results derived via simulation indicate that the algorithm of Wang et al. produces slightly fewer rejections than does MIRA although to avoids denials resulting from either concentrating and distributing vertices.

The crucial step in relation to the efficiency of these algorithms is the use of max-flow algorithms in the identification of critical links. MIRA [6] uses maximum flow in conjunction with the minimum cut set to identify

critical links whereas the WSC algorithm [7] detects critical links from their contribution to the maximum flow. Executing such max-flow algorithms, however, is prohibitively expensive for large networks. The *Edmonds-Karp* algorithm, which is the most well known implementation of the *Ford-Fulkerson*'s max-flow computation method [8], uses a breadth search to find the augmenting paths it has a complexity of $O(V \cdot E^2)$. For networks such as the Internet which has an average node degree of about 3.5 [9], the number of links is significantly greater than that of vertices, so that the *Edmonds-Karp* algorithm performs very inefficiently in dense graphs.

Goldberg's algorithm, which is the fastest known max-flow algorithm still involves a prohibitively high complexity for problems involving large networks. It has a complexity of $O(\min(V^{2/3}, E^{1/2}) \cdot E \cdot \log(V^2/E) \cdot \log(U))$ for networks with $|V|$ vertices and $|E|$ links, with capacities in the interval [1,U] [10]. Other approaches that do not use max-flow computation are thus necessary to reduce the complexity of minimum interference algorithms.

III. Light Minimum Interference Routing Algorithm

In this section, the Light Minimum Interference Routing (LMIR) algorithm is presented. The main characteristic of this algorithm is that its computational complexity is lower than that of other existing minimum interference algorithms. This reduced computational complexity is achieved by avoiding numerous executions of max-flow algorithms for the identification of critical links.

The central idea of the LMIR algorithm is the selection of links for the establishment of LSPs which exert the least impact on the maximum flow between other SD pairs by identifying the links with the smallest available capacity as critical links. Such links are close to saturation and are likely to be included in any minimum cut set of the network. To identify critical links, the LMIR algorithm identifies the paths of least capacity, since all critical links belong to those paths, although a single least-capacity path may involve more than one critical link.

The first step in the LMIR algorithm involves the identification of K least capacity paths with the use of the LowestCapacities algorithm, a variation of Dijkstra algorithm. This LowestCapacities algorithm, compares the flow at a node v with the smaller of the two values represented by the possible flow of the link (u, v) and the actual flow at the node u. The algorithm stores the information about the distance between a source node and any other node as a function of the number of hops between these nodes ($di[]$), as well as the minimum capacity of all paths between that source node and all other nodes ($F[]$), and the precedence vector involving in reaching a specific destination ($\pi[]$).

Initially, all non-source nodes are attributed an infinitely large value for both flow and distance whereas these values are considered to be zero at the source node; the node prior to the source node is not considered(Steps 2 to 4).

The first adjacent node v is then analyzed. The flow value at the node v ($F[v]$) is established to be the lower of the two values representing either 1) the current value of $F[s]$ or 2) the flow value of the link (Step 11). The distance (number of hops) is then increased by one (Step 12) with s becoming the new predecessor (Step 13). Additional adjacent nodes are queued according to increasing distance values for future adjacency analysis (Step

14). As nodes are removed from the queue using procedure *extract_min(Q)* and their adjacent nodes processed the algorithm progresses.

For each step of the LowestCapacities algorithm, the metric values associated with the adjacent nodes of any specific source node are updated; this updating occurs whenever the value of the flow which begins at the source node (s) and ends at the adjacent node (v) and has passed through the target node (u) and is either smaller than the value of $F[v]$ or is the same, but located at distance at least one hop less than that between the source and the destination distance, $di[u] < di[v] - 1$. Removing the links with the least capacity from the network makes it possible to identify other paths with critical links so that paths with ever-increasing capacities will be identified.

Algorithm 1 LowestCapacities

1: **for all** $(v \in |V|)$ **do**
2: $F[v], di[v] = \infty$
3: $\pi[v] = NIL$
4: F[s], di[s] = 0.0
5: $Q \leftarrow s$
6: **while** (Q) **do**
7: $u \leftarrow extract\_min(Q)$
8: **for all** $v \in ADJ[u]$ **do**
9: $\gamma = min[c(u, v), F[u]]$
10: **if** $[(\gamma < F[v]) \vee ((\gamma == F[v]) \wedge (di[u] < di[v]))]$ **then**
11: $F[v] = \gamma$
12: $di[v] = di[u] + 1$
13: $\pi[v] = u$
14: $Q \leftarrow Q \bigcup v$

Once the paths with least capacity are identified, they are assigned weights according to the following formula (Step 2):

$$w(u, v) = \sum_{(s',d') \in P} \frac{f_G^{(s',d')}}{c_r(u, v)}, \qquad (u, v) \in E_r, \tag{3}$$

where $c_r(u, v)$ is the residual capacity of link (u, v) and $f_G^{(s',d')}$ is the flow between s' and d', the path with the lowest capacity.

The weight values assigned to the links of the paths found in Step 1 are thus inversely proportional to the residual capacity. The next step involves the removal of links with a residual capacity smaller than that required (Step 3). Finally, the Dijkstra algorithm is executed using $w(u, v)$ as weights to select non-a-less critical links (Step 4). The LMIR algorithm is presented as Algorithm 2.

The complexity of the LMIR algorithm can be analyzed as follows. Step 6 of *LowestCapacities* algorithm is executed $|V|$ times, since all vertices are visited. Step 8 is also executed $|V|$, times resulting in a complexity of $O(V^2)$. The function *extract_min(Q)* involves a complexity of $O(E_f)$ in the worst possible case, as would happen

Algorithm 2 LMIR

INPUT

A residual graph $G_r = (V, E_r)$ and $r(s, d, bw)$ and a request for bw bandwidth units between pair (s, d).

OUTPUT

A path (route) with bw bandwidth units connecting s to d.

LMIR

1: Find K least capacity paths $\forall(s', d') \in P$ (applying LowestCapacities algorithm).
2: Compute weights according to equation (3) for all the links belonging to the paths identified.
3: Eliminate all links with residual capacity smaller than bw.
4: Execute the *Dijkstra* algorithm using $w(u, v)$ as weights.
5: Create an LSP connecting s to d and update the capacity of the links.

when the queue of vertices with non-visited adjacent vertices is implemented as a linked list. Thus, the complexity of the LMIR algorithm is, $O(V^2 + E_f) = O(V^2)$ [8].

Since each pair requires K executions of the LowestCapacities algorithm, the complexity of the LMIR algorithm for identifying critical links is $O(K \cdot V^2) = O(V^2)$, whereas in the other existing minimum interference algorithms the complexity involves the performance of a max-flow algorithm which has a complexity of $O(\min(V^{2/3}, E_f^{1/2}) \cdot E \cdot \log(V^2/E_f) \cdot \log(U))$. Thus, the following theorem can then be affirmed:

Theorem 1: The computational complexity of the LMIR algorithm is upperbounded by the complexity of the MIRA algorithm.

Proof: See Appendix I. ■

The number of least capacity paths considered by the LMIR algorithm has a tremendous impact on its performance. Although it is not possible to determine an optimum value for the number of least capacity paths to be identified, an upper bound for that value can be established (Theorem 2).

Theorem 2: An upper bound, $K(\theta)$, for the number of least capacity paths to be identified by the LMIR algorithm is given by:

$$K(\theta) = d(u), \text{ if } \theta^{(u,v)} = \sum_{\forall i|(u,i)\in E(G)} c(u, i), \tag{4}$$

or

$$K(\theta) = d(v), \text{ if } \theta^{(u,v)} = \sum_{\forall i|(i,v)\in E(G)} c(i, v), \tag{5}$$

or

$$K(\theta) = \left\lceil \frac{\theta^{(u,v)}}{\omega^{(u,v)}} \right\rceil. \tag{6}$$

where: $\omega_{(G)}^{(u,v)}$ is the flow along the path of least capacity between u and v in G, and $\theta_{(G)}^{(u,v)}$ is the maximum flow between them.

Proof:

See Appendix II ■

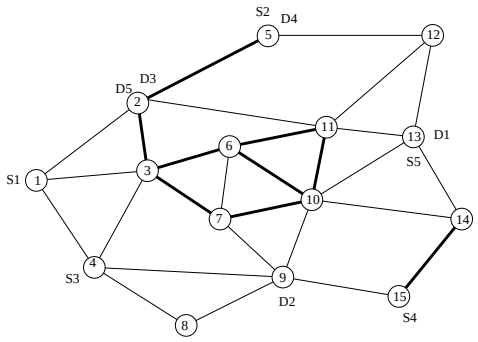


Fig. 2. Topology used in simulations for small networks.

IV. A COMPARISON BETWEEN THE LMIR ALGORITHM AND OTHER EXISTING MINIMUM INTERFERENCE ALGORITHMS

Simulation experiments were used to compare the performance of the LMIR algorithm with that of both the MIRA and WSC algorithms. Although the Minimum Hop Algorithm (MHA) and the Widest Shortest Path algorithm (WSP) do not take interference into account, they were also included in the simulation experiments so that the benefi ts of a non-interference approach can be assessed. MHA is actually the Dijkstra algorithm; the WSP algorithm identifi es either the path with maximum capacity in the network or that with the least number of links.

Both, large and small networks were considered in this study. The small network used in both [6] and [7], was utilized so that a comparison could be made to previously published results, while the topology for the large networks was randomly generated.

The topology used in the small network experiments as well as the source-destination pairs are shown in Figure 2. The lighter bi-directional links represent a capacity of 1,200 bandwidth units each, while darker ones have capacity of 4,800 bandwidth units each, corresponding to transmission rates of OC-12 and OC-48, respectively. Eight thousand requests among the fi ve SD pairs were randomly generated using a uniform distribution in the interval $[1, 4]$ with long-lived LSPs assumed so that once accepted, resources are held until the end of the experiments.

Figure 3 shows the reduction in total bandwidth available as a function of the number of requests. It can be seen that the minimum interference routing algorithms produce lower reductions than do WSP and MHA algrithms, since the latter two do not consider the criticality of link in choosing a paths.

Up to 3,000 requests, the results of the LMIR and MIRA algorithms are equivalent both resulting in less bandwidth reduction than the WSC algorithm, but between 3,000and 5,000 requests, the decrease in reduction is greater for the LMIR algorithm.

The importance of the non-interference approach can be seen in the large reduction resulting from the application of the WSP and MHA algorithms. This reduction is a direct consequence of the unnecessary allocation of critical links, greatly increasing the impact on the maximum flow.

Figure 4 shows the number of requests rejected as a function of the number of requests arriving. The WSP

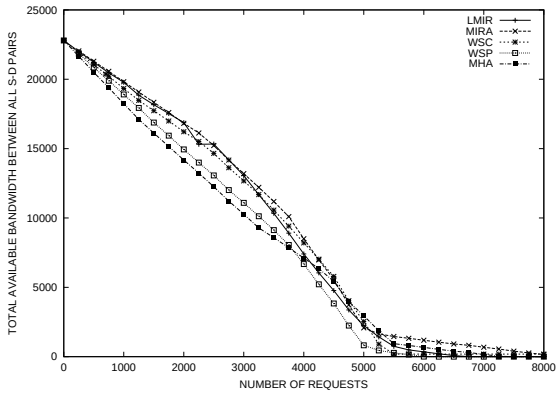


Fig. 3. Total bandwidth for all network source-destination pairs.

algorithm starts rejecting connections only after the arrival of 5,000 requests. However, since it does not take link criticality into consideration, it may choose paths that include critical links of various pairs, thus leading to saturation. The LMIR and MIRA algorithms result in the lowest number of requests rejected, which proves the benefits of the non-interference approach.

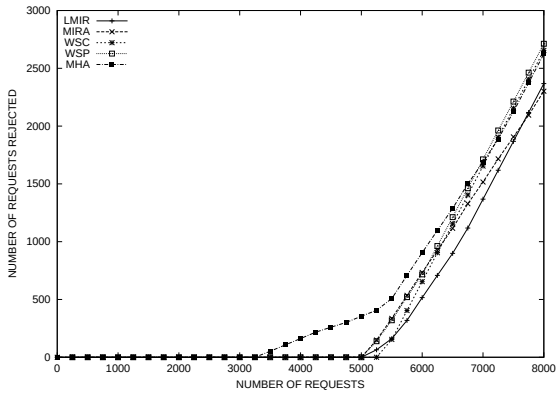


Fig. 4. Number of rejections.

The two previous examples have illustrated the performance of different algorithms in relation to the total flow in a network. Another interesting question is how these algorithms influence the flow of an individual SD pair. To shed some light on this question, the pair $(S1, D1)$ was chosen as the object of this analysis. Figure 5 shows the reduction in maximum flow of this source-destination pair as a function the other SD pairs. It can be observed that the use of the MHA and WSP algorithms leads to a reduction in the maximum flow as soon as the first request arrives. Network saturation when using the MHA algorithm, is reached immediately after the arrival of the $2,000^{th}$ request. From there on, the path is saturated. The WSP algorithm takes a somewhat longer period to reach saturation since when one path is approaching saturation it chooses other high-capacity.

The minimum interference algorithms do not provide any interference under low-load conditions (up to the arrival of the $2,500^{th}$ request), but both the LMIR and MIRA algorithms produce less interference than the WSC

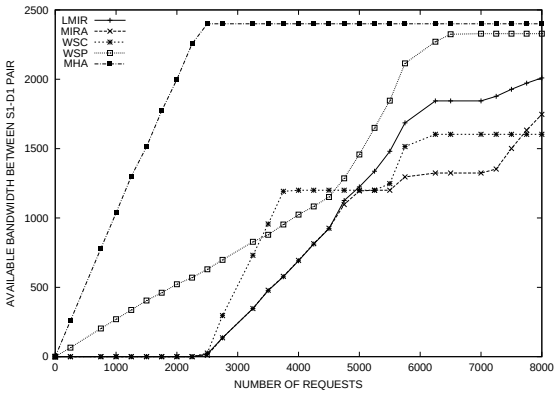


Fig. 5. Effects on pair $(S1, D1)$.

algorithm for as many as 5.000 requests, although in the long term the use of the WSC algorithm results in a less of a reduction in the maximum flow of the pair than do the LMIR and MIRA algorithms which produce similar interference up to the $5,000^{th}$ request since they choose the same set of critical links. After the arrival of the $5,000^{th}$ request, however, the MIRA algorithm identifies fewer paths not interfering with the maximum flow of pair $(S1, D1)$ than does the LMIR algorithm.

The choice of the number of least capacity paths to be identified in Step 2 is the key to the performance of the LMIR algorithm since the selection of a value for K which differs significantly from the number of links in the minimum cut set, may lead to the misidentification of critical links and the attribution of high weight values to them.. Figure 6 shows the number of requests rejected as a function of the choice of K. This shows that the minimum number of requests rejected are when $K = 5$. Increasing the value of K does not lead to any improvement, yet makes the identification of critical links difficult.

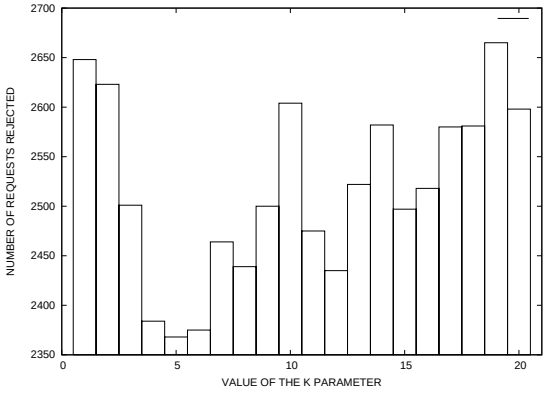


Fig. 6. Influence of parameter K in the number of rejected connections.

For the speed of performance of an even large number of requests, the algorithms were tested for 8,000 requests measured with the Linux *time* command in an Intel Celeron machines with a 1.2 GHz clock, and 256 MB of RAM. Table I shows the values measured.

TABLE I

EXECUTION TIMES FOR 8,000 REQUESTS.

	WSC	MIRA	LMIR $K=1$	LMIR $K=2$	LMIR $K=3$	LMIR $K=4$	LMIR $K=5$	LMIR $K=6$
Time	7.23s	7.24s	5.017s	5.67s	6.54s	6.94s	7.01s	7.14s

Both the WSP and MHA algorithms require execution times of aproximately 4 seconds. Although these are the fastest results, the algorithms are not satisfactory because they lead to an excessive number of rejections. As expected, the time required to the LMIR algorithm is less than that for both the WSC and MIRA algorithms, although the numerical results produced are the same for all three. Actually, there is a trade off between precision and execution time involving the number of least capacity paths chosen. The lower the value of K, the faster is the execution time. However, the results tend to differ signifi cantly from the optimum. Selecting a larger number of least capacity paths than the optimum only increases the execution time.

Large networks simulations were also conducted with the LMIR algorithm using networks with 30, 40, and 50 vertices. For these large networks, both sparse and dense, only minimum interference algorithms were tested. Networks are modeled as non-oriented graphs in which each link has a non-negative capacity. For this simulations, network topologies were created using Waxman's method [9], [11]. In this method, the probability of the existence of link between u and v is given by

$$P(u,v) = \alpha e^{-d/(\beta L)},$$

where $0 < \alpha, \beta \leq 1$ are model parameters, d is the Euclidean distance between u and v, and L is the maximum Euclidean distance between any two vertices of the graph. Links are bidirectional and an equal number of choice for each bandwidth size (1,200 or 4,800) was made.

LSPs are again assumed to be long-lived and 30,000 requests were and only generated among the fi ve source-destination pairs with the bandwidth requested uniformly distributed in the interval $[1, 4]$. Scenarios with low between 0 and $10,000$, mean between $10,000$ and $20,000$ and high between $20,000$ and $30,000$ loads were considered.

A. Dense 30-vertex networks

Figure 7 shows the maximum flow reduction for the selected source-destination pairs for dense 30-vertex networks, with all algorithms evaluated performing at a similar level. This performance worsens under high loads, since links belonging to the minimum cut set may be used to establish LSPs. With high loads, 70% of the requests are rejected (Figure 8). Both LMIR and MIRA algorithms block roughly the same number of requests, fewer than those blocked by the WSC algorithm. Only towards the end of the simulation does the LMIR algorithm produce slightly lower blocking probabilities than does the MIRA algorithm.

Under high loads, only a small number of paths having available capacities are subject to allocation. The difference

in the performance of the algorithms can be attributed to the way they identify critical links, which leads to different approachs to saturation.

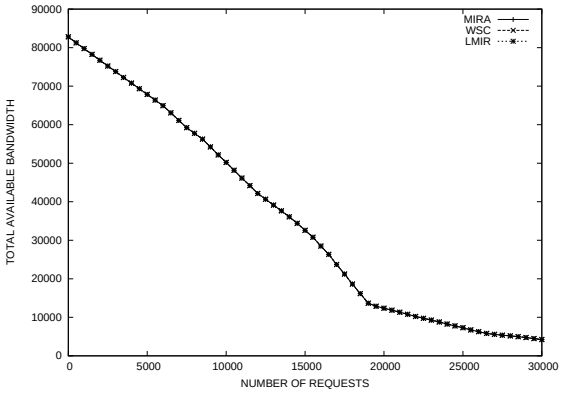


Fig. 7. Total bandwidth (dense 30-vertex networks).

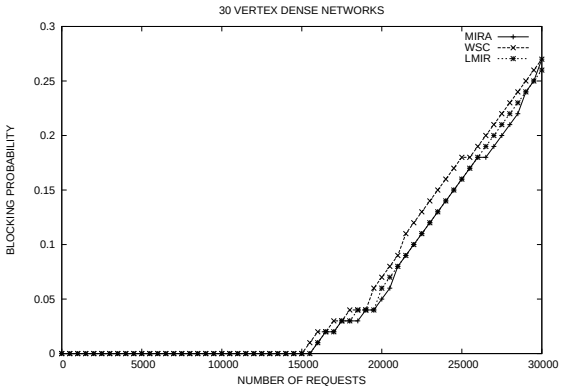


Fig. 8. Blocking Probability (dense 30-vertex networks).

B. Sparse 30-vertex networks

Figure 9 illustrates the maximum flow reduction between source-destination pairs for sparse networks with 30 vertices. The behavior of the three algorithms (WSC, MIRA, and LMIR) is very similar whatever the load condition.

This happens because of the reduced number of paths existing for each source-destination pair, as this obviously reduces the number of alternative paths for each algorithm. Moreover, the blocking probability values (Figure 10) become almost indistinguishable under high loads. However, under low and medium loads, the LMIR algorithm produces blocking probabilities which are 0.17 and 0.1 lower than those produced by the WSC and MIRA algorithms, respectively.

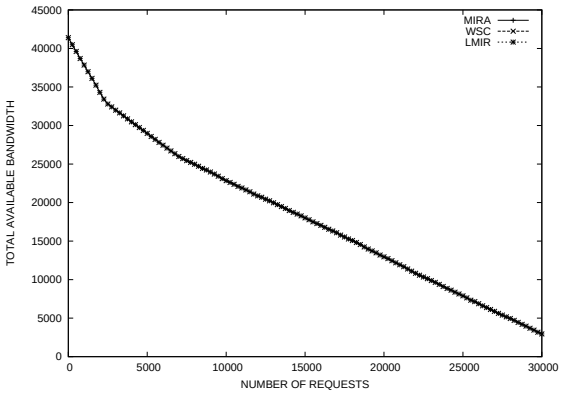


Fig. 9. Total bandwidth (sparse 30-vertex networks).

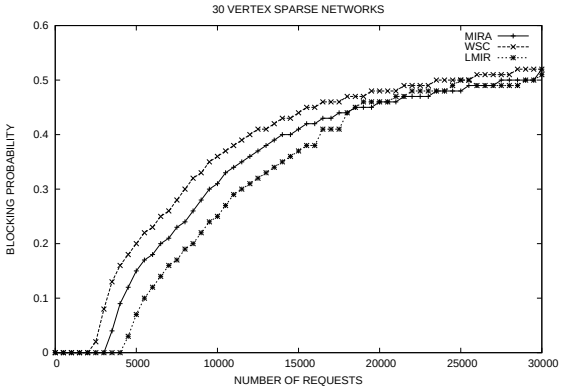


Fig. 10. Blocking Probability (sparse 30-vertex networks).

C. Dense 50-vertex networks

The dense 50-vertex networks used here have 544 links which implies on a large number of paths to choose from. As a consequence, saturation was not observed in the simulation experiments. To achieve saturation in such networks, all of the links are assigned a capacity of 1,200 bandwidth units.

The behavior of the WSC, MIRA, and LMIR algorithms was found to be quite similar for up to 20,000 requests (Figure 11). For medium loads the LMIR algorithm produced the lowest probability of blocking (Figure 12), whereas for light loads the probability of blocking was null for all three algirihms, and for high loads their performance were almost indistinguishable.

The number of rejections in these sparse 50-vertex networks is substantially lower than in the dense networks or those with a smaller number of available links, due to the large number of paths available to establish LSPs.

D. Sparse 50-vertex networks

Sparse networks with 50 vertices (Figure 13)are subject to a slower reduction in the maximum flow when the LMIR algorithm was used than when the WSC and MIRA. With 30 nodes, saturation is reached rapidly due to the

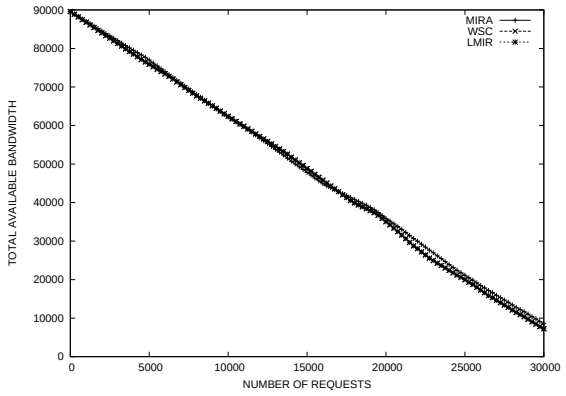


Fig. 11. Total bandwidth (dense 50-vertex networks).

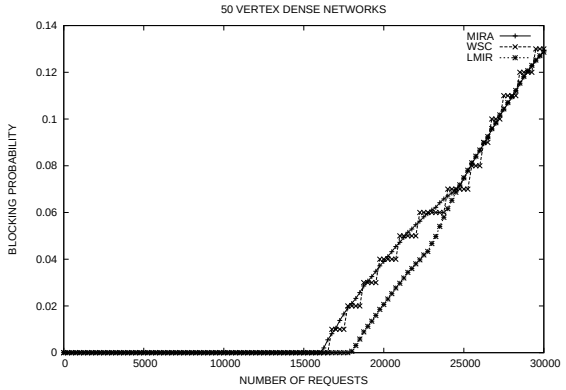


Fig. 12. Blocking Probability (50 vertices dense networks).

saturation of critical links. This rapid saturation means that the maximum flow reduction is more intense in sparse than in dense networks with the same number of nodes.

The use of links belonging to the minimum cut set implies the absence of alternative paths and consequently leads to a high number of rejections (Figure 14). Nonetheless, the choice of paths pursued by the LMIR algorithm produces the lowest number of rejections of requests, whereas the WSC algorithm produces the highest number.

Figure 14 shows the probability of blocking in sparse networks with 50 vertices. The critical links are used intensively due to the lack of alternative paths, which leads to a high number of request rejection. The lowest blocking probability is produced by the LMIR algorithm.

The relative gain in execution time for the minimum interference algorithms studied here was determined and is shown in Table II. The highest value (produced by either the MIRA or WSC algorithms) is used as a standard (0% gain). These results show that the LMIR algorithm is the fastest, with reductions of up to 39% when five least capacity paths were involved, although this performance deteriorates for a larger number of paths. In all the simulations, the most propitious was five as this resulted in the fastest execution time while producing blocking probability values similar to those obtained using the MIRA and WSC algorithm.

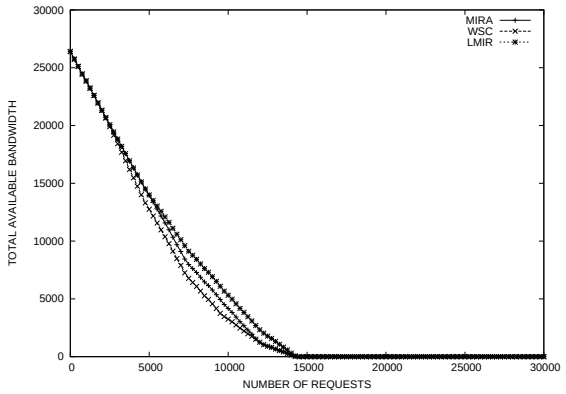


Fig. 13. Total bandwidth (sparse 50-vertex networks).

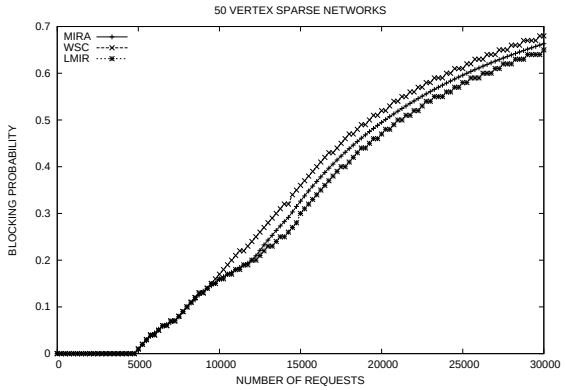


Fig. 14. Number of requests rejected (sparse 50-vertex networks).

V. Conclusions

In this paper, a new minimum interference routing algorithm which does not use max-flow algorithms for critical link identification, was presented. This LMIR algorithm uses a modified Dijkstra algorithm to identify paths with the least capacity and these paths are used to identify the critical links. Weights are then assigned to the links of these paths, and the shortest path is identified. The number of critical paths to be identified when using the LMIR algorithm impact on its performance. The best results in simulation experiments were obtained when this was limited to five.

The results obtained using the LMIR algorithm are similar to those obtained with the MIRA and WSC algorithms. Moreover, for large dense networks, it produced the lowest probability of blocking as well as minimal reductions in maximum flow. Furthermore, the LMIR algorithm involves some 39% less computational time than the MIRA and WSC algirithms. In general, this novel algorithm, thus, seems to be the best candidate for adoption in on-line procedures for the establishment of LSP's in MPLS networks.

Acknowledgment

This work was partially sponsored by the Brazilian National Council for Research (CNPq).

TABLE II

RELATIVE EXECUTION TIME GAINS FOR INDIVIDUAL CONNECTIONS

	50 (D)	50 (E)	40 (D)	40 (E)	30 (D)	30 (E)
Standard	6.836×10^{-3}	5.683×10^{-3}	4.851×10^{-3}	5.740×10^{-3}	4.079×10^{-3}	6.477×10^{-3}
MIRA	0.00	1%	1%	0.00	0.00	0.00
WSC	2%	0.0	0.00	0%	2%	20%
LMIR K =5	39%	17%	13%	36%	2%	28%
LMIR K =6	30%	12%	5%	23%	-1%	22%
LMIR K =7	28%	5%	-7%	18%	-6%	19%
LMIR K =8	27%	3%	-9%	-1%	-11%	11%
LMIR K =9	10%	0%	-22%	-6%	-20%	4%
LMIR K =10	2%	-6%	-24%	-6%	-29%	-2%

REFERENCES

[1] G. B. Figueiredo, "Algoritmos de Roteamento com Interferência Mínima," Master's thesis, Instituto de Computaç ˜ao - UNICAMP. http://www.ic.unicamp.br/˜ gustavo/dissertacao, 2003.

[2] G. B. Figueiredo, J. A. S. Monteiro, N. L. S. da Fonseca, and A. A. A. Rocha, "Dynamic Sizing of Label Switched Paths in MPLS Networks," in *IEEE International Telecommunications Symposium*, 2002, pp. 593–598.

[3] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Co., 1979.

[4] S. Chen and K. Nahrstedt, "QoS Routing for Next-Generation Networks," *IEEE Network*, Dezembro 1998.

[5] G. B. Figueiredo, N. L. S. da Fonseca, and J. A. S. Monteiro, "A Minimum Interference Routing Algorithm," in *To appear in proceedings of IEEE International Conference on Communications, Paris -France*, 2004.

[6] M. S. Kodialam and T. V. Lakshman, "Minimum Interference Routing with Applications to MPLS Traffi c Engineering," in *INFOCOM (2)*, 2000, pp. 884–893.

[7] B. Wang, X. Su, and C. Chen, "A New Bandwidth Guaranteed Routing Algorithm for MPLS Traffi c Engineering," in *Proceedings of IEEE International Conference on Communications*, 2002, pp. 1001–1005.

[8] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*. MIT Press and McGraw Hill, 1990.

[9] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *IEEE Infocom*, vol. 2. San Francisco, CA: IEEE, March 1996, pp. 594–602.

[10] A. V. G. e Satish Rao, "Beyond the fbw decomposition barrier," in *J. ACM 45 (5)*, 1998, pp. 783–797.

[11] B. M. Waxman, "Routing of multipoint connections," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1617–1622, 1988.

[12] D. B. West, *Introduction to Graph Theory*, P. Hall, Ed. Prentice Hall, 1996.

APPENDIX I

This appendix furnishes proofs for the theorems presented in the paper.

Theorem 1: The computational complexity of the LMIR algorithm is upper bounded by the complexity of the MIRA algorithm.

Proof:

Since Steps 2 through 5 of the LMIR algorithm (see Algorithm 2) are also the same a those executed by the MIRA and WSC algorithms and since the complexity of these steps is a function of the procedure used to identify the critical links, it is sufficient to show that the complexity of the LowestCapacities algorithm is less than that of Goldberg's algorithm, the least complex max-flow algorithm known.

The proof is carried out for both sparse and dense graphs. For **dense graphs** it is assumed that $|E_f| = |V|^2$. Goldberg's algorithm has the following complexity:

$$O((V^{2/3}) \cdot V^2 \cdot \log(V^2/V^2) \cdot \log(U)),$$

while the *LowestCapacities* algorithm has the complexity of $O(V^2)$. Assuming that some of the terms involved in $\log(V^2/V^2)$ have been omitted (since the complexity cannot be zero), and given the fact that $\log(V^2/V^2) \cdot \log(U) > 1$, it follows that

$$\text{Goldberg's } \textit{max-flow} = O(V^{2/3} \cdot V^2 = V^{8/3}), \text{ and}$$

$$(V^{8/3} > V^2) = \text{LowestCapacities algorithm.}$$

Hence, $V^2 = O(V^{8/3}) \Rightarrow$ LowestCapacities algorithm $= O($Goldberg's *max-flow*$)$ □.

For **sparse graphs** it is assumed that $|E_f| = |V|$. In this case, Q (the queue of non visited nodes) can be implemented as a heap [8]. Therefore,

$$\textit{LowestCapacities algorithm} = O(V \cdot \log V),$$

since the step of *extract_min(Q)* evidences a complexity of $O(\log V)$. Assuming that $\log(V^2/E_f) \cdot \log(U) > 1$, using the first two factors of $O(\min(V^{2/3}, E_f^{1/2}) \cdot E_f \cdot \log(V^2/E_f) \cdot \log(U))$, it can be shown that

$$\text{Goldberg's max-flow} = V^{1/2} \cdot V, \text{ and}$$

$$\textit{LowestCapacities algorithm} = O(V \cdot \log V).$$

Discarding V in both algorithms and taking the $\log$ gives:

$$\textit{LowestCapacities algorithm} = \log(\log V), \text{ and}$$

$$\text{Goldberg's max-flow} = \log V^{1/2} = 1/2 \cdot \log V.$$

Hence, $\log(\log(V)) = O(1/2 \cdot \log V) \Rightarrow$ *LowestCapacities algorithm* $= O($Goldberg's max-flow$)$.□

This shows that the critical links detection with the LMIR algorithm has a lower computational complexity than does any other minimum interference algorithm using max-flow algorithms. ■

APPENDIX II

Theorem 2: An upper bound, $K(\theta)$, for the number of least capacity paths identified by the LMIR algorithm is given by:

$$K(\theta) = d(u), \text{ if } \theta^{(u,v)} = \sum_{\forall i|(u,i)\in E(G)} c(u,i), \tag{7}$$

or

$$K(\theta) = d(v), \text{ if } \theta^{(u,v)} = \sum_{\forall i|(i,v)\in E(G)} c(i,v), \tag{8}$$

or

$$K(\theta) = \left\lceil \frac{\theta^{(u,v)}}{\omega^{(u,v)}} \right\rceil. \tag{9}$$

where: $\omega_{(G)}^{(u,v)}$ is the flow along the path of least capacity between u and v in G, and $\theta_{(G)}^{(u,v)}$ is the maximum flow between u and v.

Proof: To find the lower bound for the number of links in the minimum cut set, it is necessary to identify the minimum number of links which forms disconnected graphs. Before showing the proof for the lower bound value, some definitions must be introduced.

Definition 1: Let $G = (V, E)$ be a graph and let $V(G)$ and $E(G)$ be its vertex and edge sets, respectively. A **disconnecting set** is a set $F \subseteq E(G)$ such that $G - F$ has more than one component. A graph is said to be k-edge-connected if every disconnecting set has at least k edges. The edge-connectivity, $\kappa'(G)$, is the minimum size of a disconnecting set [12].

Definition 2: Two paths from u to v are internally disjointed if they have no common internal vertices (edges) [12].

Theorem 3: Let $\lambda(u, v)$ be the number of u, v-internally disjointed paths (or just disjointed paths); then

$$\lambda(u, v) \leq min\{d(u), d(v)\},$$

where $d(u)$ and $d(v)$ are the degrees of u and v, respectively.

Proof: Let P and P' be two internally disjointed paths between u and v. Now, suppose by contradiction that $\lambda(u, v) > \min\{d(u), d(v)\}$. If $\lambda(u, v) > \min\{d(u), d(v)\}$, this implies that at least one edge incident to u or v was included in both of the internally disjointed paths, P and P'. However, if P and P' have an edge in common they are not internally disjointed thus contradicting the hypothesis. ■

The following theorem establishes a relationship between the cardinality of a set of disconnecting links and the number of disjointed paths of a graph.

Menger's theorem [12] uses these concepts of disconnecting sets and disjointed paths:

Theorem (Menger-1927) 4: Let u and v be two vertices of a graph G and $(u, v) \notin E(G)$. Then, $\max(\kappa'(G)) = \min(\lambda(u, v))$.

Proof: See [12]. ■

The relation between Theorem 3 and Menger's Theorem suggests an upper bound for K, since it indicates the maximum number of edges a minimum size disconnecting set can have.

However, as we shall see, this is not true. Figure 15 illustrates a graph with only two disjointed paths but with four edges in the minimum cut set. This graph can be transformed into an equivalent one to which Menger's Theorem can be applied. The transformation involves changing each edge with a capacity x into x edges with a capacity one.

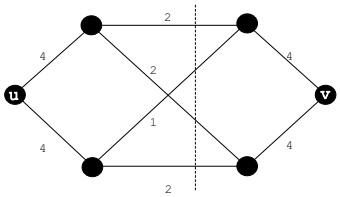


Fig. 15. Number of disjoint paths less than the number of edges in the min cut set.

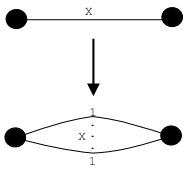


Fig. 16. Transformation in the original network.

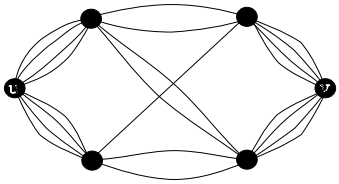


Fig. 17. Transformed network for Figure 15.

Thus, a new theorem relating maximum flow and Menger's theorem (Theorem 4) is presented:

Theorem 5:

$$\lambda(G') \geq \theta^{uv} = \min C(S,T) \geq \kappa'(G'),$$

where $\lambda(G')$ is the number of internally disjointed paths between u and v, θ^{uv} is the maximum flow between u and v, $cap(S,T)$ represents the capacity of a cut (S,T) such that $u \in S$ and $v \in T$, and $\kappa'(G)$ is the minimum size of a disconnecting set of G.

Proof: See [12]. ■

Theorem 6: Let $f_G^{(u,v)}$ be a flow passing through a path of least capacity between u and v in G, thus $n = f_G^{(u,v)} = \sum_{i=1}^{n} f_{G'}^{(u,v)}$.

Proof: In G', each edge has a capacity of 1 unit of flow. Thus, in order to send n units of flow, we have to use n distinct paths. Furthermore, the capacity of a path in G is limited by the flow of the edge of least capacity on that path. Since G' is built from G, including an edge with a capacity of R in G is equivalent to using R edges with the same endpoints in G'. Thus the equality holds. ■

Case 1:

$$\theta^{(u,v)} = \sum_{\forall i | (u,i) \in E(G)} c(u,i).$$

The Max-flow Min-cut theorem [12] implies that the maximum network flow is bounded by the minimum capacity of a source/sink cut. Thus if

$$\theta^{(u,v)} = \sum_{\forall i | (u,i) \in E(G)} c(u,i),$$

the maximum network flow is bounded by the capacities of the edges incident to u.

Case 2:

$$\theta^{(u,v)} = \sum_{\forall i|(i,v)\in E(G)} c(i,v).$$

This can be proved using reasoning analogous to that for Case 1.

Case 3:

Let

$$n = \theta^{(u,v)}_{(G')} = \theta^{(u,v)}_{(G)},$$

It is known that all paths have capacity of 1 in G'. Therefore,

$$n = \theta^{(u,v)}_{(G)} = n \cdot f^{(u,v)}_{G'} = \sum_{i=1}^{n} f^{(u,v)}_{G'},$$

if ω and $K(\theta)$ are chosen so that $\omega \cdot K(\theta) \leq n$, then

$$n = \theta^{(u,v)}_{(G)} \geq K(\theta) \cdot \sum_{i=1}^{\omega} f^{(u,v)}_{G'}. \tag{10}$$

The application of Theorem 6 in Equation 10 gives the following:

$$K(\theta) \cdot \omega^{(u,v)}_{(G)} \leq \theta^{(u,v)}_{(G)}. \tag{11}$$

Since in G both $\theta^{(u,v)}_{(G)}$ and $\omega^{(u,v)}_{(G)}$ are known, we can define $K(\theta)$ as

$$K(\theta) = \left\lceil \frac{\theta^{(u,v)}}{\omega^{(u,v)}} \right\rceil.$$

■

Algoritmo Robusto de Aprendizaje para el Modelo Mezcla de Expertos

Romina D. Torres[1,2]
Director: Dr. Héctor Allende; Correferente: Dr. Horst von Brand ; Externo: Dr. Max Chacón.
[1]Universidad Técnica Federico Santa María Dept. de Informática,
Av. España 1680, Casilla 110-V, Valparaíso-Chile;

[2] Motorola Valparaíso; Global Software Group
romina.torres@motorola.com

Abstract

El Modelo de Mezcla de Expertos (ME) es un tipo de Redes Neuronales Artificiales Modulares (MANN) especialmente adecuadas cuando el espacio de entrada se encuentra estratificado. La arquitectura está compuesta por diferentes módulos: redes expertas que compiten por aprender diferentes aspectos de un problema y una red de agregación que arbitra la competencia y aprende a asignar diferentes regiones del espacio de datos a diferentes expertos locales cuya topología parece ser la más apropiada. La regla de aprendizaje combina aspectos competitivos y asociativos y está diseñada para favorecer la competencia entre expertos locales, que permiten dividir el espacio 'automáticamente' en subregiones manejadas en lo posible por un único experto local.

El aprendizaje del modelo ME puede ser tratado como un problema de estimación de parámetros que maximizan la verosimilitud, donde el algoritmo de Máxima Expectación desacopla el proceso de estimación en una manera que calza con la estructura modular de la arquitectura ME.

Sin embargo, cuando los datos están expuestos a datos atípicos, el modelo es afectado debido a que el algoritmo es sensible a estas desviaciones obteniendo un bajo rendimiento. En esta tesis se propone robustificar el algoritmo EM para el modelo ME, obteniendo un algoritmo elegante, eficiente, de rápida convergencia debido a que aprovecha la modularidad del modelo (baja interferencia destructiva), y a la vez es insensible a los datos atípicos (acotando el impacto de ellos en la obtención de los estimadores pero sin eliminarlos). Para ésto se utiliza una generalización del estimador máximo verosímil conocido como M-estimadores.

En la fase de prueba se seleccionan problemas reales y con presencia de datos atípicos pertenecientes a la serie de problemas estándares DELVE y PROBEN1, para mostrar que el algoritmo Robusto de Máxima Expectación para Mezcla de Expertos (REM-ME) muestra mejoras significativas con respecto a los métodos clásicos.

Palabras Claves: Redes Neuronales Artificiales Modulares, Modelos de Mezcla, Modelo Mezcla de Expertos, M-estimadores, Algoritmo de Máxima Expectación.

1 Introducción

El cerebro es un sistema altamente complejo debido a que está compuesto de un gran número de componentes que interactúan entre sí. Estudios en Biología y Psicología han descubierto la arquitectura modular del cerebro, pero no han llegado a un acuerdo en el número de módulos, la naturaleza de su interacción o la manera en que ellos se desarrollan. Estos módulos se especializan en ciertas funciones de acuerdo a sus propiedades estructurales, presentando alta cohesión. Una pregunta válida es si la definición de cuál módulo realiza qué función es acorde al material genético o en base a la experiencia.

Un creciente número de investigadores cree que lo que realmente determina las propiedades funcionales de las distintas regiones del cerebro, es un proceso dependiente de la experiencia. Esta teoría combina dos grandes nociones:

- Debido a las diferentes propiedades estructurales de las regiones del cerebro, existe una correspondencia entre sus estructuras y las funciones que son capaces de realizar. Aunque diferentes regiones puedan realizar una misma tarea o puedan adaptarse para realizarla, algunas serán más eficientes que otras.

- Diferentes regiones del cerebro compiten por la habilidad de realizar un conjunto de tareas. Basados en la primera noción se puede establecer que la competencia está sesgada, ya que cada región tiende a ganar la competencia en aquellas funciones en las cuales su estructura es la más adecuada.

Jacobs et al. en [JJNH91a], proponen una arquitectura de redes neuronales, conocida como Modelo Mezcla de Expertos (ME) que implementan las nociones anteriores utilizando modelos conexionistas. Esta arquitectura es ilustrada en la Figura 1, y consiste en dos tipos de redes neuronales: redes expertas y una red de agregación (conocida como *gating* en la literatura inglesa). Las redes expertas compiten por aprender de los patrones de entrenamiento, mientras la red de agregación media esta competencia.

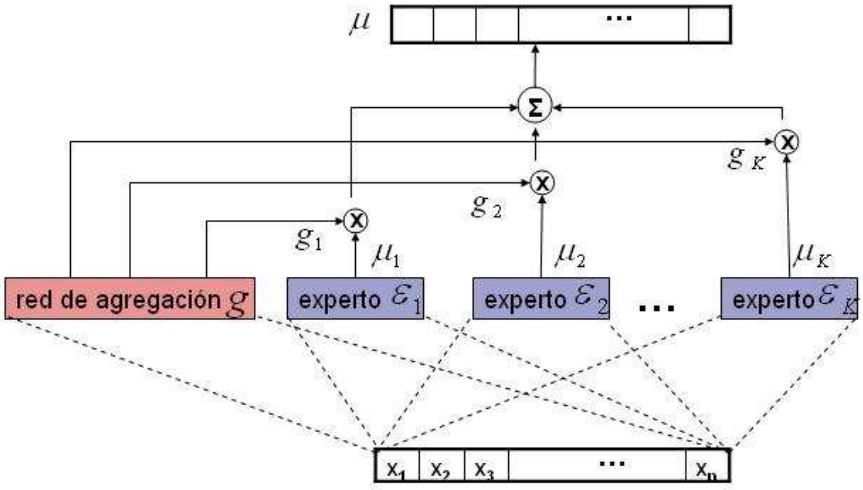


Figure 1: *Modelo Mezcla de Expertos(ME)*: consiste en un conjunto de redes expertas y una red de agregación. Los expertos compiten por aprender diferentes aspectos del problema y la red de agregación es la mediadora de esta competencia.

El modelo ME es un caso particular de una arquitectura compuesta por un conjunto de redes que tienen la capacidad de inhibirse entre sí, suprimiendo la salida de otras redes. La fuerza inhibitoria de la arquitectura depende del contexto del problema, es decir del valor actual de los patrones. Aunque en el cerebro, no existe evidencia física de una red de agregación, al finalizar el entrenamiento, aquellas redes que ganan la competencia, suprimen fuertemente la salida de las redes perdedoras (con pobre rendimiento). Existen diversos autores que a partir de los resultados obtenidos por este tipo de arquitectura sugieren que en el cerebro los módulos neuronales interactúan entre sí inhibitoriamente.

El problema de aprendizaje de la arquitectura ME puede ser tratado como un problema de estimación de parámetros. Un enfoque común es aplicar el método de gradiente descendente para obtener los estimadores de máxima verosimilitud (ML). En [JJ91], [JJB91] y [JJ92], resultados empíricos revelan que, aunque el enfoque del gradiente es exitoso en encontrar valores razonables de los parámetros en problemas particulares, la razón de convergencia no fue significante mayor que el obtenido al utilizar el método de gradiente en arquitecturas feedforward.

Un enfoque alternativo es presentado en [JJ94], donde se introduce el algoritmo de Máxima Expectación (EM) para la arquitectura Mezcla de Expertos que mostró tomar ventaja de la modularidad de la arquitectura. El algoritmo EM es un algoritmo de propósito general para obtener estimadores de máxima verosimilitud en una amplia variedad de situaciones mejor descritas como problemas de *datos incompletos*. Aunque no existan datos perdidos u otra forma de incompletitud en los datos, casi siempre es posible reformular un problema dentro de un marco EM. La formulación actual del algoritmo EM [DLR77], considera una variedad de ejemplos y condiciones especiales para su convergencia.

En esta tesis, se propone utilizar el algoritmo EM para estimar los parámetros del modelo ME de manera de aprovechar la ventaja de la estructura modular de la arquitectura. Se mostrará que cuando se presentan datos atípicos en la muestra (valores aberrantes que se alejan fuertemente del modelo subyacente), denominados outliers en la literatura inglesa, el supuesto distribucional del comportamiento entre los datos de entrada y de salida ya no es completamente válido. Tales datos atípicos son usualmente eliminados utilizando un filtro basado en un umbral para la chi-cuadrada antes de aplicar el algoritmo EM. Sin embargo, encontrar el umbral adecuado es difícil y usualmente será arbitrario. Consecuentemente con ésto, información 'necesaria' podría ser rechazada como datos atípicos. Por ejemplo, en problemas de clasificación,

una clase con pocas muestras sería difícil de identificar y la teoría de filtros podría asumir que estas muestras constituyen outliers estadísticos, eliminando información que podría ser la semilla para una nueva clase.

Como una alternativa a las técnicas habituales de preprocesamiento de los datos, estudios de robustez, en especial en la etapa de entrenamiento de redes feedforward han sido propuestas en varios trabajos (ver [CM94], [AMS01], [AMS02]) donde se ha mostrado la efectividad de utilizar algoritmos que acotan la influencia de los datos atípicos sin llegar a eliminarlos.

Debido a que la arquitectura ME es sensible a la presencia de datos atípicos, el objetivo de esta tesis es construir un algoritmo EM Robusto para la estimación de los parámetros mediante la maximización de la verosimilitud del modelo neuronal ME utilizando M-estimadores, de manera tal que la influencia de los datos atípicos es acotada.

2 Alcance y Contribución de esta Tesis

El objetivo de esta tesis es desarrollar un algoritmo de aprendizaje robusto para el modelo Mezcla de Expertos, que le permita ser insensible a la presencia de datos atípicos y que tome ventaja de la modularidad del modelo (en cuanto a rapidez de convergencia).

En base a este objetivo general, en esta tesis se realizan las siguientes hipótesis:

- Elección del Modelo ME: cuando se tienen tareas de distinta naturaleza que se desean modelar por una única red se produce un problema en el aprendizaje conocido como 'interferencia destructiva'. La elección más adecuada en estos casos es seleccionar el modelo ME donde distintas redes modelan distintas tareas, y por tanto no interfieren entre sí durante el aprendizaje.
- Se obtienen mejoras significativas en el rendimiento obtenido por el modelo ME en el conjunto de prueba al obtener los parámetros mediante un algoritmo insensible a los datos atípicos.
- Se reduce la complejidad del modelo. Cuando el algoritmo no considera un tratamiento para los datos atípicos, es altamente probable que el número de expertos del modelo ME aumente debido a que el modelo trata de modelar los datos atípicos con expertos adicionales. La introducción de robustez permite modelar la real tendencia de los datos independiente del ruido que presenten.
- El modelo ME converge al menos un orden de magnitud más rápido si se utiliza durante el aprendizaje el algoritmo EM en vez de un algoritmo basado en el gradiente, debido a que tomará ventaja de la modularidad del modelo.
- Cuando los conjuntos de datos presentan datos atípicos, el modelo ME con aprendizaje robusto presentará mejoras significativas en el rendimiento.

Esta tesis tiene una contribución especial al mundo científico debido a que el modelo Mezcla de Expertos desde su aparición a comienzos de los años 90, [JJNH91b], ha sido exitosamente utilizado en problemas que involucran diferentes fuentes de información. En este tipo de problemas el modelo ME ha mostrado ser superior a una única red neuronal en cuanto a su rendimiento, su capacidad de generalización y su razón de convergencia. Lamentablemente, cuando los datos son ruidosos, presentando datos atípicos (usualmente los conjuntos de datos reales presentan estas características), el modelo ME disminuye su capacidad de generalización, y disminuye su rapidez de convergencia, siendo necesario usualmente aumentar el número de expertos de la mezcla, y por lo tanto aumenta el número de parámetros a estimar, ver [TSAM02], [TSAM03] y [ATSM03]. Esta tesis presenta una posible solución a este problema, robustificando el algoritmo de aprendizaje, adquiriendo por tanto insensibilidad frente a los datos estadísticos atípicos, sin rechazarlos (como lo haría una teoría de filtros) pero acotando su influencia, permitiendo que el modelo extraiga la información relevante que ellos pudiesen enmascarar.

Es importante mencionar, que durante el desarrollo de esta Tesis se generaron las siguientes publicaciones indexadas en Congresos internacionales: [TSAM03] y [ATSM03]. Un tercer artículo ha sido aceptado para su pronta publicación [AMST04].

3 Mezcla de Expertos

Cuando el algoritmo de gradiente descendente es utilizado para entrenar una red multicapa para realizar diferentes subtareas, generalmente aparecerá un efecto de interferencia que generará un proceso de aprendizaje lento y una pobre capacidad de generalización. Si se conoce a priori que el conjunto de casos de entrenamiento, puede ser naturalmente dividido en

subconjuntos que corresponden a cada una de las distintas subtareas, la interferencia puede ser reducida utilizando un sistema compuesto de distintas redes expertas más una red de agregación, que decide cuáles de los expertos deberían ser usados para cada observación.

Cuando se tiene un espacio de entrada estratificado en diferentes subregiones, mediante el aprendizaje se debe inferir que existirán distintos mapas en cada subregión. Aunque una única red homogénea podría ser aplicada a este problema, se espera que las tareas sean más fáciles de realizar si se asignasen diferentes redes expertas a cada una de las subregiones, y el criterio de asignación de un experto a una subregión fuese realizado por una red extra, denominada red de agregación, que recibe como información durante el aprendizaje, el patrón de entrada y el rendimiento relativo de los expertos para ese patrón.

La arquitectura (ME), [JJ99], está compuesta de K redes expertas, cada una de las cuales resuelve un problema de aproximación de función sobre una región local del espacio de entrada. El conjunto de datos es denominado Υ, que está formado por duplas compuestas de vectores de entrada $\underline{x} \in \mathbb{R}^n$ y vectores de salida $\underline{y} \in \mathbb{R}^m$. En esta sección, utilizaremos la versión extendida del conjunto de datos, es decir su descomposición explícita en vectores de entrada y salida.

A cada una de las redes expertas del modelo ME, denominado ε_j, se le asocia un modelo probabilístico como sigue:

$$P(\underline{y}|\underline{x}, \underline{\theta}_j, \Sigma_j), \qquad j = 1, \ldots, K, \tag{1}$$

donde $\underline{\theta}_j$ es el vector de los parámetros de la red experta j-ésima y Σ_j es la matriz de covarianza de los residuos generados entre la diferencia del dato verdadero y la salida del experto. Por motivos de simplicidad se asume que estas densidades de probabilidad pertenecen a la familia exponencial, restringiendo el análisis del presente trabajo a modelos Gaussianos.

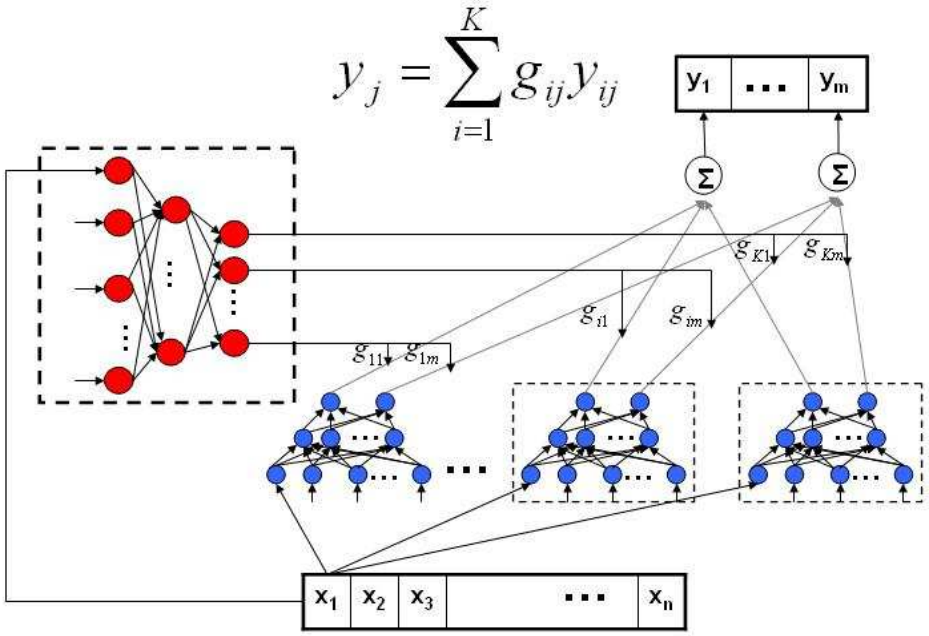


Figure 2: *Arquitectura Mezcla de Expertos(ME)*: La arquitectura ME consiste en un conjunto de redes expertas y una red de agregación. Los expertos compiten por aprender diferentes aspectos del problema y la red de agregación es la mediadora de la competencia.

Considerando la figura 2, cada experto ε_j (red j-ésima) genera como salida un vector de parámetros μ_j, con

$$\underline{\mu}_j = E_P[\underline{y}|\underline{x}, \underline{\theta}_j] = \underline{f}_j(\underline{x}, \underline{\theta}_j), \quad j = 1, \ldots, K, \tag{2}$$

donde $\underline{\mu}_j$ es el parámetro de localización del modelo $P(\underline{y}|\underline{x}, \underline{\theta}_j, \Sigma_j)$(en este caso es simplemente la media). En esta tesis se asumirá que los $\underline{f}_j$ son lineales en los parámetros y que la componente de no linealidad del modelo es entregada por la red de agregación, debido a que cada salida, g_i, probabilidad condicional de los datos y del rendimiento de cada experto.

Cada red experta tiene asociada una matriz de covarianza, Σ_j, no singular, por lo tanto el modelo probabilístico para el experto ε_j, puede ser escrito como $N(f_j(\underline{x},\underline{\theta}_j),\Sigma_j)$, o explícitamente

$$P(\underline{y}|\underline{x},\underline{\theta}_j,\Sigma_j) = \frac{1}{(2\pi)^{\frac{m}{2}}\|\Sigma_j\|^{\frac{1}{2}}} \exp\left\{-\tfrac{1}{2}[\underline{y}-\underline{f}_j(\underline{x},\underline{\theta}_j)]^T\Sigma_j^{-1}[\underline{y}-\underline{f}_j(\underline{x},\underline{\theta}_j)]\right\} \tag{3}$$

Diferentes redes expertas son apropiadas en diferentes regiones del espacio de entrada, por lo tanto la arquitectura determina estocásticamente para una entrada $\underline{x}$, cuál experto o mezcla de expertos es más apropiada para producir la salida deseada. Para este fin, la arquitectura ME también utiliza una red auxiliar conocida como la *red de agregación* (gating) que particiona el espacio de entrada en diferentes regiones correspondientes a diferentes redes o mezcla de redes expertas. Esto se logra asignando un vector de probabilidad $[g_1,g_2,\ldots,g_K]^T$ a cada punto del espacio de entrada. En particular, la red de agregación implementa una función parametrizada $\xi(\underline{x},\underline{\theta}_0)$ y una función normalizadora $\underline{g}(\xi)$. La función ξ es un mapa desde $\mathbb{R}^n$ a $\mathbb{R}^K$, para cada elemento del vector de parámetros $\underline{\theta}_0$, y la función $\underline{g}$ es un mapa de $\mathbb{R}^K$ a $\mathbb{R}^K$. Una forma particular de $\underline{g}$ es la función softmax,

$$g_j = g_j(\underline{x},\underline{\theta}_0) = \frac{e^{\xi_j(\underline{x},\underline{\theta}_0)}}{\sum_{i=1}^K e^{\xi_i(\underline{x},\underline{\theta}_0)}}, \qquad j=1,\ldots,K. \tag{4}$$

Notar que esta definición implica que g_j es no-negativa, $g_j \geq 0$, y $\sum_{i=1}^K g_i = 1$.

Una interpretación probabilística es ver g_j como una superficie discriminante en un problema de clasificación y por lo tanto a la red de agregación como un sistema *clasificador* que mapea una entrada $\underline{x}$ a la probabilidad de que un experto o una mezcla de expertos sea capaz de generar la salida deseada (basado sólo en el conocimiento de $\underline{x}$).

Es importante hacer notar, que aunque se seleccione una estructura lineal para las redes expertas, el espacio es dividido suavemente gracias a la forma de las salidas de la red de agregación, siendo éstas las que entregan la componente de no linealidad al modelo, permitiendo utilizar arquitecturas más simples para las redes expertas con salidas lineales en vez de sigmoidales sin degradar el rendimiento obtenido por el modelo.

Se asume que los datos de entrenamiento son generados de acuerdo al siguiente modelo de probabilidad. Asumiendo que para un $\underline{x}$ dado, un experto ε_j es seleccionado con probabilidad $P(\varepsilon_j|\underline{x},\underline{\theta}_0) = g_j(\underline{x},\underline{\theta}_0)$. Una salida $\underline{y}$ es entonces escogida con probabilidad $P(\underline{y}|\underline{x},\underline{\theta}_j,\Sigma_j)$. Por lo tanto, la probabilidad total de generar $\underline{y}$ dada la entrada $\underline{x}$ es dado por la siguiente densidad de mezcla,

$$P(\underline{y}|\underline{x}) = \sum_{j=1}^K P(\varepsilon_j|\underline{x},\underline{\theta}_0)P(\underline{y}|\underline{x},\underline{\theta}_j,\Sigma_j) = \sum_{j=1}^K g_j(\underline{x},\underline{\theta}_0)P(\underline{y}|\underline{x},\underline{\theta}_j,\Sigma_j) \tag{5}$$

Las *redes expertas* modelan los distintos procesos que generan los datos, y la *red de agregación* modela la decisión de utilizar uno de esos diferentes procesos. Cuando el aprendizaje es extremadamente competitivo, la salida $\underline{y}$ es generada por un único experto.

Se asume que el conjunto de entrenamiento $\Upsilon = \{(\underline{x}^{(t)},\underline{y}^{(t)})\}_{t=1}^N$ es generado de la siguiente manera: dada la elección de la entrada $\underline{x}$, un experto ε_j es escogido con probabilidad $P(\varepsilon_i|\underline{x},\underline{\theta}_0^*)$ (donde el exponente '*' distingue los valores de los parámetros reales del modelo). Dada la elección del experto ε_j y dada la entrada $\underline{x}$, se asume que la salida deseada $\underline{y}$ es generada con probabilidad $P(\underline{y}|\underline{x},\underline{\theta}_j^*,\Sigma_j)$.

4 Algoritmo de Aprendizaje basado en el gradiente

Para desarrollar un algoritmo con el fin de estimar los parámetros de la arquitectura de una mezcla de expertos, se puede utilizar el principio de máxima verosimilitud (ML). Esto significa que se deben escoger los parámetros para los cuáles la probabilidad del conjunto de entrenamiento dado los parámetros (función conocida como verosimilitud) es máxima.

Dado un conjunto de N muestras independientes e idénticamente distribuidas, $\Upsilon = \{(\underline{x}^{(t)},\underline{y}^{(t)})\}_{t=1}^N$ la verosimilitud correspondiente a una mezcla de K-componentes es

$$L(\underline{\Theta},\Upsilon) = P(\{\underline{y}^{(t)}\}_1^N|\{\underline{x}^{(t)}\}_1^N) = \prod_{t=1}^N P(\underline{y}^{(t)}|\underline{x}^{(t)}) = \prod_{t=1}^N\sum_{j=1}^K g_j(\underline{x}^{(t)},\underline{\theta}_0)P(\underline{y}^{(t)}|\underline{x}^{(t)},\underline{\theta}_j,\Sigma_j) \tag{6}$$

El problema de aprendizaje de la arquitectura ME es tratado como un problema de encontrar los estimadores de los parámetros $\underline{\theta}_0$, $\underline{\theta}_j$ y Σ_j que maximicen la función de verosimilitud $L(\underline{\Theta},\Upsilon)$. Como es común en Estadística, es más conveniente trabajar con el logaritmo de la verosimilitud que con ella misma. Tomando el logaritmo del producto de N densidades lleva a la siguiente ecuación,

$$l(\underline{\Theta},\Upsilon) = \ln\prod_{t=1}^{N}\sum_{j=1}^{K} g_j(\underline{x}^{(t)},\underline{\theta}_0)P(\underline{y}^{(t)}|\underline{x}^{(t)},\underline{\theta}_j,\Sigma_j) = \sum_{t=1}^{N}\ln\sum_{j=1}^{K} g_j(\underline{x}^{(t)},\underline{\theta}_0)P(\underline{y}^{(t)}|\underline{x}^{(t)},\underline{\theta}_j,\Sigma_j) \tag{7}$$

donde $\underline{\Theta} = [\underline{\theta}_0,\underline{\theta}_1,\ldots,\underline{\theta}_K,\Sigma_1,\Sigma_2,\ldots,\Sigma_K]^T$.

Para obtener los parámetros de las redes expertas y la red de agregación, el logaritmo de la función de la verosimilitud $l(\underline{\Theta},\Upsilon)$, debe ser maximizada a través de un proceso de aprendizaje del modelo. En este trabajo el gradiente ascendente es aplicado a la ecuación (7).

Como se nombró anteriormente, la red de agregación es considerada lineal, $\underline{\xi} = \theta_0^T\underline{x}$, sus salidas son obtenidas luego de aplicar la función softmax. La distribución condicional es $P(\underline{y}^{(t)}|\underline{x}^{(t)},\underline{\theta}_j,\Sigma_j) = \frac{1}{(2\pi)^{\frac{m}{2}}}\exp\left\{-\frac{(\underline{y}-\underline{\mu}_i)^T(\underline{y}-\underline{\mu}_i)}{2}\right\}$. Calculando el gradiente de $l = l(\underline{\Theta},\Upsilon)$ con respecto a $\underline{\xi}_j$ y $\underline{\mu}_j$ se obtiene que

$$\frac{\partial l}{\partial \underline{\mu}_j} = \sum_{t=1}^{N} h_j^{(t)}\frac{\partial}{\partial \underline{\mu}_j}\ln P(\underline{y}^{(t)}|\underline{x}^{(t)},\underline{\theta}_j,\Sigma_j) \tag{8}$$

y

$$\frac{\partial l}{\partial \underline{\xi}_j} = \sum_t (h_j^{(t)} - g_j^{(t)}) \tag{9}$$

donde $h_j^{(t)}$ es definido como $P(\varepsilon_j|\underline{x}^{(t)},\underline{y}^{(t)})$. Utilizando la regla de Bayes:

$$P(\varepsilon_j|\underline{x}^{(t)},\underline{y}^{(t)}) = \frac{P(\varepsilon_j|\underline{x}^{(t)})P(\underline{y}^{(t)}|\underline{x}^{(t)},\varepsilon_j)}{\sum_i P(\varepsilon_i|\underline{x}^{(t)})P(\underline{y}^{(t)}|\underline{x}^{(t)},\varepsilon_i)} \tag{10}$$

Esto sugiere que, $h_j^{(t)}$ está definido como la *probabilidad a posteriori* del experto j-ésimo, condicionado a la entrada $\underline{x}^{(t)}$ y a la salida $\underline{y}^{(t)}$. Similarmente, la probabilidad $g_j^{(t)}$ puede ser interpretada como la *probabilidad a priori* $P(\varepsilon_j|\underline{x}^{(t)})$ del experto j-ésimo dado solamente la entrada $\underline{x}^{(t)}$. Dada estas definiciones, la ecuación (9) tiene una interpretación natural de mover las probabilidades a priori hacia las probabilidades a posteriori.

Un caso especial es una arquitectura en que las redes expertas y la red de agregación son lineales y con densidad de probabilidad Gaussiana con $\Sigma = I$. Por lo tanto la ecuación (8) y (9) nos permiten escribir explícitamentelas la regla de actualización de los parámetros para la red de agregación (11):

$$\Delta\underline{\theta}_0 = \underline{\theta}_0^{k+1} - \underline{\theta}_0^k = \alpha\sum_t (h_j^{(t)} - g_j^{(t)})\underline{x}^{(t)^T} \tag{11}$$

Para las redes expertas (12) se obtiene que:

$$\Delta\underline{\theta}_j = \alpha\sum_{t=1}^{N} h_j^{(t)}(\underline{y}^{(t)} - \underline{\mu}_j)\underline{x}^{(t)^T} \tag{12}$$

donde α es la razón de aprendizaje. La expresión de la probabilidad a posteriori en el caso Gaussiano viene dada por:

$$h_j^{(t)} = \frac{g_j^{(t)}\exp\{-\frac{1}{2}(\underline{y}^{(t)} - \underline{\mu}_j^{(t)})^T(\underline{y}^{(t)} - \underline{\mu}_j^{(t)})\}}{\sum_i g_i^{(t)}\exp\{-\frac{1}{2}(\underline{y}^{(t)} - \underline{\mu}_i^{(t)})^T(\underline{y}^{(t)} - \underline{\mu}_i^{(t)})\}} \tag{13}$$

Esto es una medida de distancia normalizada que refleja las magnitudes relativas de los residuales $\underline{y}^{(t)} - \underline{\mu}_i^{(t)}$. Si los residuales para el experto ε_j son pequeños relativo a los otros, entonces $h_j^{(t)}$ es grande. Los $h_j^{(t)}$ son positivos y su suma por cada $\underline{x}^{(t)}$ es uno; ésto implica que la responsabilidad es distribuida entre los expertos en una manera competitiva.

5 Algoritmo de Máxima Expectación para el Modelo Mezcla de Expertos

Jacobs et al. en [JJ91] basados en pruebas empíricas realizadas en su investigación, revelaron que aunque el enfoque del gradiente encontraba exitosamente los valores de los parámetros para problemas particulares, la razón de convergencia no era mejor que la obtenida utilizando el enfoque del gradiente en redes multicapas. La razón es que el gradiente no parecía tomar ventaja de la modularidad de la arquitectura.

Un método alternativo al enfoque del gradiente fue propuesto por Jordan y Jacobs en [JJ94], quienes introdujeron el algoritmo de Máxima Expectación (EM) ([DLR77]) para la arquitectura Mezcla de Expertos. Para esta arquitectura el algoritmo EM desacopla el proceso de estimación en una manera que ajusta bien con la estructura modular de la arquitectura.

EM está basado en la idea de resolver una sucesión de problemas simplificados que son obtenidos por aumentar las variables observadas originalmente con un conjunto adicional de variables denominadas *escondidas*. .

Un conjunto de datos observados Υ, es aumentado por un conjunto Υ_{perd}, denominado conjunto de variables *perdidas* o *escondidas*. Considérese el problema de máxima verosimilitud para un conjunto de *datos completos*, $\mathbb{Z} = \{\Upsilon, \Upsilon_{perd}\}$. Las variables perdidas se escogen de tal manera que el logaritmo de la máxima verosimilitud de los *datos completos*, dado por $l_c(\underline{\Theta}, \mathbb{Z}) = \log P(\Upsilon, \Upsilon_{perd}|\underline{\Theta})$, es fácil de maximizar con respecto a $\underline{\Theta}$. El modelo de probabilidad $P(\Upsilon, \Upsilon_{perd}|\underline{\Theta})$ debe ser escogido de tal manera que su distribución marginal a través de Υ, denominado en este contexto como la verosimilitud de los *datos incompletos* es la verosimilitud original

$$P(\Upsilon|\underline{\Theta}) = \int_{\Upsilon_{perd}} P(\Upsilon, \Upsilon_{perd}|\underline{\Theta}) d\Upsilon_{perd} \tag{14}$$

Debido a que la verosimilitud es una función de las variables aleatorias perdidas, no se puede trabajar directamente con la verosimilitud de los *datos completos*. Es necesario promediar Υ_{perd}, es decir maximizar el *valor esperado* del logaritmo de la verosimilitud de los *datos completos* $E_{\Upsilon_{perd}}[\log P(\Upsilon, \Upsilon_{perd}|\underline{\Theta})|\Upsilon, \Theta]$.

El algoritmo EM es un algoritmo iterativo que consta de dos pasos:

- El paso de expectación (E) que calcula el valor esperado condicional del logaritmo de la verosimilitud

$$Q(\underline{\Theta}|\underline{\Theta}^{(k)}) \quad = \int_{\Upsilon_{perd}} P(\Upsilon_{perd}|\Upsilon, \underline{\Theta}^{(k)}) \log P(\mathbb{Z}|\underline{\Theta}) d\Upsilon_{perd} \tag{15}$$

  donde $\underline{\Theta}^{(k)}$ es el valor esperado del vector de parámetros en la iteración k.

- El paso de maximización (M) que calcula

$$\underline{\Theta}^{(k+1)} = \arg\max_{\underline{\Theta}} Q(\underline{\Theta}|\underline{\Theta}^{(k)}) \tag{16}$$

El paso M escoge un valor para el vector de parámetros que aumenta la función Q, es decir el valor esperado del logaritmo de la verosimilitud de los *datos completos*. Dempster et al. [DLR77] muestran que una iteración del algoritmo EM también aumenta el logaritmo de la verosimilitud original l. Es decir,

$$l(\underline{\Theta}^{(k+1)}; \Upsilon) \geq l(\underline{\Theta}^{(k)}; \Upsilon) \tag{17}$$

Eso quiere decir que la función de verosimilitud l crece monóticamente de acuerdo a la secuencias de parámetros estimados generados por el algoritmo EM. La solución del paso M puede ser obtenido analíticamente y en aquellos casos en que no sea posible, será necesario realizar un procedimiento iterativo interno para optimizar Q.

Aunque en la práctica, frecuentemente la solución del paso M existe en una forma cerrada, existen veces en que no es factible encontrar el valor de $\underline{\Theta}$ que maximiza globalmente la función $Q(\underline{\Theta}, \underline{\Theta}^{(k)})$. Para tales situaciones, Dempster et al. [DLR77], definieron un algoritmo EM generalizado (algoritmo GEM) para el cuál el paso M requiere escoger $\underline{\Theta}^{(k+1)}$ tal que:

$$Q(\underline{\Theta}^{(k+1)}, \underline{\Theta}^{(k)}) \geq Q(\underline{\Theta}^{(k)}, \underline{\Theta}^{(k)}) \tag{18}$$

Esto significa escoger un $\underline{\Theta}^{(k+1)}$ que aumenta la función Q sobre su valor en $\underline{\Theta} = \underline{\Theta}^{(k)}$, en vez de maximizarlo sobre todo $\underline{\Theta} \in \Omega$. Como se puede ver en [MP01] (sección 3.3), la condición anterior en $\underline{\Theta}^{(k+1)}$ es suficiente para asegurar que

$$L(\underline{\Theta}^{(k+1)}) \geq L(\underline{\Theta}^{(k)}) \tag{19}$$

En este caso, la verosimilitud $L(\underline{\Theta},\Upsilon)$ no decrece después de una iteración GEM, y así una secuencia GEM de valores de verosimilitud debe converger si existe un límite superior.

Para la arquitectura ME se escogen los datos perdidos como variables aleatorias indicadoras $\Upsilon_{perd} = \{\gamma^{(t)}_{perd_j}, j = 1,\ldots,K,\ t = 1,\ldots,N\}$ con

$$\gamma^{(t)}_{perd_j} = \begin{cases} 1, & \text{si } \underline{y}^{(t)} \text{ es generado por el modelo } j-esimo. \\ 0, & e.t.o.c. \end{cases} \tag{20}$$

y

$$\sum_{j=1}^{K} \gamma^{(t)}_{perd_j} = 1,\ para\ cada\ t \tag{21}$$

Asumiendo que la distribución de los *datos completos* es dada por

$$P(\mathbb{Z}|\underline{\Theta}) = \prod_{t=1}^{N}\prod_{j=1}^{K}[g_j(\underline{x}^{(t)},\underline{\theta}_0)P(\underline{y}^{(t)}|\underline{x}^{(t)},\underline{\theta}_j,\Sigma_j)]^{\gamma^{(t)}_{perd_j}} \tag{22}$$

ecuación que satisface (14).

De (15) se obtiene que

$$\begin{aligned} Q(\underline{\Theta}|\underline{\Theta}^{(k)}) &= E_{\Upsilon_{perd}}\{\ln P(\mathbb{Z}|\underline{\Theta})|\Upsilon,\underline{\Theta}^{(k)}\} \\ &= \Sigma_{t=1}^{N}\Sigma_{j=1}^{K} h_j^{(k)}(t)\ln g_j(\underline{x}^{(t)},\underline{\theta}_0) + \Sigma_{t=1}^{N} h_1^{(k)}(t)\ln P(\underline{y}^{(t)}|\underline{x}^{(t)},\underline{\theta}_1,\Sigma_1) \\ &+ \ldots + \Sigma_{t=1}^{N} h_K^{(k)}(t)\ln P(\underline{y}^{(t)}|\underline{x}^{(t)},\underline{\theta}_K,\Sigma_K) \end{aligned} \tag{23}$$

donde

$$\begin{aligned} h_j^{(k)}(t) &= E[\gamma^{(t)}_{perd_j}|\Upsilon,\underline{\Theta}^{(k)}] = P(j|\underline{x}^{(t)},\underline{y}^{(t)}) \\ &= \frac{g_j(\underline{x}^{(t)},\underline{\theta}_0^{(k)})P(\underline{y}^{(t)}|\underline{x}^{(t)},\underline{\theta}_j^{(k)},\Sigma_j^{(k)})}{\Sigma_{i=1}^{K} g_i(\underline{x}^{(t)},\underline{\theta}_0^{(k)})P(\underline{y}^{(t)}|\underline{x}^{(t)},\underline{\theta}_i^{(k)},\Sigma_i^{(k)})}, \end{aligned} \tag{24}$$

donde $P(j|\underline{x}^{(t)},\underline{y}^{(t)})$ denota la probabilidad de que el par $\{\underline{x}^{(t)},\underline{y}^{(t)}\}$ haya sido generado por el modelo de probabilidad $j-esimo$. Notar que siempte $h_j^{(k)}(t) > 0$.

La implementación del paso M basándose en las ecuaciones (5), (4) y (23), es obtenida

$$\frac{\partial Q}{\partial \underline{\theta}_0} = \Sigma_{t=1}^{N}\Sigma_{j=1}^{K}[h_j^{(k)}(t) - g_j(\underline{x}^{(t)},\underline{\theta}_0)]\frac{\partial \underline{\xi}_j}{\partial \underline{\theta}_0} \tag{25}$$

$$\frac{\partial Q}{\partial \underline{\theta}_j} = \Sigma_{t=1}^{N} h_j^{(k)}(t)\frac{\partial \underline{f}_j^T(\underline{x}^{(t)},\underline{\theta}_j,\Sigma_j)}{\partial \underline{\theta}_j}\Sigma_j^{-1}[\underline{y}^{(t)} - \underline{f}_j(\underline{x}^{(t)},\underline{\theta}_j,\Sigma_j)], \tag{26}$$

y

$$\frac{\partial Q}{\partial \Sigma_j} = -\tfrac{1}{2}\Sigma_{t=1}^{N} h_j^{(k)}(t)\Sigma_j^{-1}\{\Sigma_j - [\underline{y}^{(t)} - \underline{f}_j(\underline{x}^{(t)},\underline{\theta}_j,\Sigma_j)][\underline{y}^{(t)} - \underline{f}_j(\underline{x}^{(t)},\underline{\theta}_j,\Sigma_j)]^T\}\Sigma_j^{-1} \tag{27}$$

Si $\frac{\partial Q}{\partial \Sigma_j}|_{\Sigma_j=\Sigma_j^{(k+1)}} = 0$, se obtiene la actualización para las matrices de covarianza

$$\Sigma_j^{(k+1)} = \frac{1}{\Sigma_{t=1}^{N} h_j^{(k)}(t)}\Sigma_{t=1}^{N} h_j^{(k)}(t)[\underline{y}^{(t)} - \underline{f}_j(\underline{x}^{(t)},\underline{\theta}_j,\Sigma_j)][\underline{y}^{(t)} - \underline{f}_j(\underline{x}^{(t)},\underline{\theta}_j,\Sigma_j)]^T \tag{28}$$

Asumiendo que el conjunto de entrenamiento Υ, es generado por un modelo de mezcla de expertos, cuando el número de muestra es suficientemente grande (relativo a la dimensión de $\underline{y}$), el espacio dado por los N vectores $[\underline{y}^{(t)} - \underline{f}_j(\underline{x}^{(t)},\underline{\theta}_j,\Sigma_j)]$ será de dimensión completa con probabilidad 1. Debido a que $h_j^{(k)}(t) > 0$, cuando el número de muestras N, es suficientemente grande las matrices $\Sigma_j^{(k+1)}$ son definidas positivas con probabilidad 1.

Haciendo $\frac{\partial Q}{\partial \underline{\theta}_j}|_{\underline{\theta}_j=\underline{\theta}_j^{(k+1)}} = 0$, se obtiene que

$$\Sigma_{t=1}^{N} h_j^{(k)}(t)\frac{\partial \underline{f}_j^T(\underline{x}^{(t)},\underline{\theta}_j,\Sigma_j)}{\partial \underline{\theta}_j}(\Sigma_j^{(k)})^{-1}[\underline{y}^{(t)} - \underline{f}_j(\underline{x}^{(t)},\underline{\theta}_j,\Sigma_j)] = 0, \tag{29}$$

lo que puede ser resuelto explícitamente dado el supuesto de que las redes expertas son lineales

$$\underline{\theta}_j^{(k+1)} = (\underline{R}_j^{(k)})^{-1}\underline{c}_j^{(k)} \tag{30}$$

donde

$$\underline{c}_j^{(k)} = \sum_{t=1}^{N} h_j^{(k)}(t) X_t (\Sigma_j^{(k)})^{-1}\underline{y}^{(t)}, \tag{31}$$

$$\underline{R}_j^{(k)} = \sum_{t=1}^{N} h_j^{(k)}(t) X_t (\Sigma_j^{(k)})^{-1} X_t^T, \tag{32}$$

Notar que $R_j^{(k)}$ es invertible con probabilidad uno cuando la muestra de tamaño N es suficientemente grande.

Finalmente, consideremos la actualización para θ_0. Jordan y Jacobs observaron que la red de agregación es una forma específica del modelo generalizado, en particular un modelo multinomial *logit*. Estos modelos pueden ser ajustados eficientemente con una variante del método de Newton conocido como *IRLS*, [Tor03].

La actualización de los parámetros de la red de agregación es obtenida como sigue. Denotando el vector gradiente en la iteración k como

$$\underline{e}_g^{(k)} = \sum_{t=1}^{N}\sum_{j=1}^{K} [h_j^{(k)}(t) - g_j(\underline{x}^{(t)}, \underline{\theta}_0^{(k)})] \frac{\partial \underline{\xi}_j}{\partial \underline{\theta}_0^{(k)}}, \tag{33}$$

y la matriz Hessiana en la iteración k como

$$\underline{R}_g^{(k)} = \sum_{t=1}^{N}\sum_{j=1}^{K} g_j(\underline{x}^{(t)}, \underline{\theta}_0^{(k)})[1 - g_j(\underline{x}^{(t)}, \underline{\theta}_0^{(k)})] \frac{\partial \underline{\xi}_j}{\partial \underline{\theta}_0} \frac{\partial \underline{\xi}_j}{\partial \underline{\theta}_0^T}. \tag{34}$$

Entonces la actualización del IRLS generalizado es dado como sigue

$$\underline{\theta}_0^{(k+1)} = \underline{\theta}_0^{(k)} + \gamma_g (\underline{R}_g^{(k)})^{-1}\underline{e}_g^{(k)} \tag{35}$$

donde γ_g es la razón de aprendizaje.
En resumen, la actualización de los parámetros del modelo ME es dado como sigue

1. (El paso E) Calcular $h_j^{(k)}(t)$ mediante la ecuación (24).

2. (El paso M) Calcular $\Sigma_j^{(k+1)}$ mediante la ecuación (28), calcular $\underline{\theta}_0^{(k+1)}$ mediante la ecuación (35), y calcular $\underline{\theta}_j^{(k+1)}$, $j = 1, \ldots, K$ mediante la ecuación (30).

6 Estimación Robusta de Parámetros para Modelo Mezcla de Expertos

La teoría de estimación de los parámetros de un modelo utilizando métodos robustos, fue desarrollada por Huber [Hub64] y [ABH$^+$72], quien la propuso para estimar de manera robusta un parámetro de localización en un contexto no-mixto. En trabajos posteriores fue extendido al caso multivariado por [Mar76]. Campbell [Cam84] derivó los M-estimadores para Modelos de Mezcla de densidades finitas, obteniendo un algoritmo tipo EM, pero con una función de ponderación, la cuál le asignaba a cada pixel una medida de tipicidad.

6.1 M-Estimadores

El proceso de estimación de los parámetros de una función de distribución, tradicionalmente es realizado usando el método de los mínimos cuadrados (LS) o bien, el método de máxima verosimilitud (MLE). La clase específica de funcionales estadísticos que se estudian en esta tesis son los M-estimadores.

Los M-estimadores fueron propuesto por Huber [Hub64] como una generalización del estimador máximo verosímil.

Un *M-estimador* T_N es definido como la solución del problema de minimización de:

$$\sum_{i=1}^{N} \rho(\Upsilon_i, T_N) = \min_{T_N}! \tag{36}$$

donde $\rho(\gamma, \underline{\Theta})$ es una función real derivable en $\underline{\Theta}$. Equivalentemente se puede definir la estimación T_N como la solución de la ecuación de estimación:

$$\sum_{i=1}^{N} \psi(\Upsilon_i, T_N) = 0 \tag{37}$$

donde $\psi(\gamma, \underline{\Theta}) = \frac{\partial \rho(\gamma, \underline{\Theta})}{\partial \underline{\Theta}}$. En particular, la elección de $\rho(\gamma, \underline{\Theta}) = -I$ corresponde al estimador MLE.

6.2 Robustificación del Algoritmo de Máxima Expectación

En esta sección, un método robusto de estimación de parámetros para el modelo Mezcla de Expertos utilizando el algoritmo EM es introducido. El objetivo final, es obtener los estimadores ML de los parámetros del modelo ME, considerando los datos atípicos, pues podrían aportar información valiosa y necesaria, pero limitando su influencia.

En esta sección se muestra el proceso de robustificación del algoritmo de máxima expectación para mezcla de expertos denominado REM-ME. El paso de expectación (paso E), dado por la ecuación (15) que calculaba una función $Q(\underline{\Theta}|\underline{\Theta}^{(k)})$ y los $h_j^{(k)}(t)$ para los K expertos es reformulado, robustificando la función $Q(\underline{\Theta}|\underline{\Theta}^{(k)})$, al introducir M-estimadores (propuesto por Huber en [Hub64]), denominándose esta nueva función $RQ(\underline{\Theta}|\underline{\Theta}^{(k)})$. En el paso de maximización del algoritmo REM-ME, se deberá maximizar la función $RQ(\underline{\Theta}|\underline{\Theta}^{(k)})$ para obtener una estimación del vector de parámetros que servirá como base para que en la próxima iteración se realice una nueva estimación de este vector, hasta su convergencia.

Como se puede ver experimentalmente en [TSAM03], cuando los datos están contaminados el rendimiento de este algoritmo es considerablemente reducido. Ésto se debe a los supuestos distribucionales que son realizados. Usualmente los datos atípicos no pueden ser ajustados por el modelo supuesto y por lo tanto éste pierde validez. Desafortunadamente, los datos reales no están libres de datos atípicos. En un modelo ME, bajo resultados empíricos, notamos que cuando existe este tipo de datos, el modelo tiende a tratarlos como nuevas clases, y que por tanto al aumentar el número de expertos, el modelo adquiere la capacidad de modelarlos, asignándole expertos a esos 'casos especiales'. Un primer problema es que la complejidad del modelo aumenta, y por lo tanto su velocidad de convergencia también. Un segundo problema es que la capacidad de modelar los datos atípicos será garantizado sólo durante el entrenamiento, perdiendo generalidad, debido a que no existe garantía acerca del comportamiento distribucional de estos datos en etapas posteriores.

Una solución a este problema es la utilización de métodos robustos que sean capaces de identificar estos datos atípicos y acoten su influencia en la estimación de los parámetros del modelo. El objetivo es no agregar mayor complejidad al modelo adicionando nuevos expertos para modelar un dato en particular, sino identificarlo, e identificar que mezcla de expertos es capaz de modelarlo, al menos aproximadamente. Ésto se logra gracias a la introducción de una función $\rho(\ln(\cdot))$ en cada experto en vez de la utilización del logaritmo natural $\ln(\cdot)$, de la siguiente manera:

$$\begin{aligned} RQ(\underline{\Theta}|\underline{\Theta}^{(k)}) \quad &= \sum_{t=1}^{N}\sum_{j=1}^{K} h_j^{(k)}(t) \ln g_j(\underline{x}^{(t)}, \underline{\theta}_0) + \\ &\sum_{j=1}^{K} \left[\sum_{t=1}^{N} h_j^{(k)}(t) \rho\left(\ln P(\underline{y}^{(t)}|\underline{x}^{(t)}, \underline{\theta}_j, \Sigma_j) \right) \right] \end{aligned} \tag{38}$$

donde $h_j^{(k)}(t)$ es dado por la ecuación (24). Nótese que, la función $\rho(\ln(\cdot))$ no es aplicado a la red de agregación.

La robustificación es introducida localmente, eso significa que cuando un nuevo patrón es presentado al experto, si el patrón está alejado de la mayoría de los datos que actualmente está modelando, la influencia introducida por este patrón en el paso M será limitada sin importar si el experto tiene o no la más alta probabilidad dada por la ecuación (24). Finalmente el modelo total será robusto por el hecho de ser no sensitivo a las observaciones outliers debido a la contribución de robustez de cada experto.

El problema es cómo escoger la correcta función $\rho(\cdot)$ para cumplir con la tarea de robustificación. En [HRRS86] algunas funciones especiales para los M-estimadores son discutidas. El objetivo es ponderar cada observación de acuerdo a la magnitud de la verosimilitud evaluada en la observación. Muestras con baja verosimilitud tienden a ser tratadas como datos atípicos y su ponderación es baja. En particular para problemas de localización, los datos que están muy alejados deben tener un impacto limitado en el algoritmo de estimación. Existen diferentes funciones que pueden ser utilizadas, y en particular en esta tesis se ha seleccionado la función de Huber dada por

$$\rho_H(z) = \begin{cases} z + \frac{1}{2}\log(2\pi) & \text{si } z \geq \frac{1}{2}(-k^2 - \log(2\pi)) \\ -k\{-2z - \log(2\pi)\}^{\frac{1}{2}} - \frac{1}{2}k^2 & \text{e.t.o.c} \end{cases} \tag{39}$$

$$\psi_H(z) = \begin{cases} 1 & \text{si } z \geq \frac{1}{2}(-k^2 - \log(2\pi)) \\ k\{-2z - \log(2\pi)\}^{-\frac{1}{2}} & \text{e.t.o.c} \end{cases} \tag{40}$$

El paso de maximización M calcula:

$$\underline{\Theta}^{(k+1)} = \arg\max_{\underline{\Theta}} RQ(\underline{\Theta}|\underline{\Theta}^{(k)}) \tag{41}$$

El paso M escoge un valor para el parámetro que aumenta la función Q; el valor esperado del logaritmo de la verosimilitud de los *datos completos*.

La implementación del paso M, basada en las ecuaciones (5), (4) y (38), es obtenida para los parámetros de la red de agregación:

$$\frac{\partial RQ}{\partial \underline{\theta}_0} = \sum_{t=1}^{N}\sum_{j=1}^{K}[h_j^{(k)}(t) - g_j(\underline{x}^{(t)}, \underline{\theta}_0)]\frac{\partial \xi_j}{\partial \underline{\theta}_0} \tag{42}$$

Sea

$$\psi_j^{(k)}(t) = \psi(\ln P(\underline{y}^{(t)}|\underline{x}_{(t)}, \underline{\theta}_j)) \tag{43}$$

Para los parámetros $\underline{\theta}_j$ de las redes expertas se obtiene que:

$$\frac{\partial RQ}{\partial \underline{\theta}_j} = \sum_{t=1}^{N} h_j^{(k)}(t)\psi_j^{(k)}(t)P(\underline{y}^{(t)}|\underline{x}^{(t)}, \underline{\theta}_j, \Sigma_j)^{-1}\frac{\partial P(\underline{y}^{(t)}|\underline{x}^{(t)}, \underline{\theta}_j, \Sigma_j)}{\partial \underline{\theta}_j} \tag{44}$$

y para los parámetros Σ_j de la redes expertas

$$\frac{\partial RQ}{\partial \Sigma_j} = \sum_{t=1}^{N} h_j^{(k)}(t)\psi_j^{(k)}(t)P(\underline{y}^{(t)}|\underline{x}^{(t)}, \underline{\theta}_j, \Sigma_j)^{-1}\frac{\partial P(\underline{y}^{(t)}|\underline{x}^{(t)}, \underline{\theta}_j, \Sigma_j)}{\partial \Sigma_j} \tag{45}$$

donde

$$\frac{\partial P(\underline{y}^{(t)}|\underline{x}^{(t)}, \underline{\theta}_j, \Sigma_j)}{\partial \underline{\theta}_j} = -2|\Sigma|^{-1/2}\varphi'(\cdot)\frac{\partial f_j(\cdot)}{\partial \underline{\theta}_j}x\Sigma^{-1}(\underline{y}^{(t)} - f_j(\cdot)) \tag{46}$$

y

$$\frac{\partial P(\underline{y}^{(t)}|\underline{x}^{(t)}, \underline{\theta}_j, \Sigma_j)}{\partial \Sigma_j} = -|\Sigma|^{-1/2}|\Sigma|^{-1}\left[\frac{1}{2}\varphi(\cdot) - \varphi'(\cdot)(y^{(t)} - f_j(\cdot))(\underline{y}^{(t)} - f_j(\cdot))^T\Sigma^{-T}\right] \tag{47}$$

donde $\varphi(\cdot) = \varphi\left\{(\underline{y}^{(t)} - f_j(\cdot))^T\Sigma_j^{-1}(\underline{y}^{(t)} - f_j(\cdot))\right\}$, $\varphi'(z) = \frac{\partial\varphi(z)}{\partial z}$ y $f_j(\cdot) = f_j(\underline{x}^{(t)}, \underline{\theta}_j)$.

Para obtener las ecuaciones de la actualización de los parámetros de la red de agregación y de las redes expertas, el método de Newton puede ser utilizado. Si se considera que la densidad $\varphi(z)$ pertenece a la familia exponencial, es decir $\varphi(z) = (2\pi)^{-m/2}\exp\{-z/2\}$, y específicamente una densidad Gaussiana, y por otro lado se considera que las redes expertas son lineales, la actualización de las matrices de covarianzas es dada por:

$$\Sigma_j^{k+1} = \frac{1}{\sum_{t=1}^{N} h_j^{(k)}(t)\psi_j^{(k)}(t)}\sum_{t=1}^{N} h_j^{(k)}(t)\psi_j^{(k)}(t)[\underline{y}^{(t)} - f_j(\cdot)][\underline{y}^{(t)} - f_j(\cdot)]^T \tag{48}$$

La actualización de los parámetros $\underline{\theta}_j$ de las redes expertas es dado por

$$\underline{\theta}_j^{(k+1)} = (\underline{R}_j^{(k)})^{-1}\underline{c}_j^{(k)} \tag{49}$$

donde

$$\underline{c}_j^{(k)} = \sum_{t=1}^{N} h_j^{(k)}(t)\psi_j^{(k)}(t)X_t(\Sigma_j^{(k)})^{-1}\underline{y}^{(t)}, \tag{50}$$

$$\underline{R}_j^{(k)} = \sum_{t=1}^{N} h_j^{(k)}(t)\psi_j^{(k)}(t)X_t(\Sigma_j^{(k)})^{-1}X_t^T, \tag{51}$$

La actualización de los parámetros de la red de agregación es obtenida mediante el algoritmo IRLS generalizado como sigue:

$$\underline{\theta}_0^{(k+1)} = \underline{\theta}_0^{(k)} + \alpha(\underline{R}_0^{(k)})^{-1}\underline{e}_0^{(k)} \tag{52}$$

donde α_0 es la razón de aprendizaje de la red de agregación y

$$\underline{e}_0^{(k)} = \sum_{t=1}^{N}\sum_{j=1}^{K}[h_j^{(k)}(t) - g_j(\underline{x}^{(t)}, \underline{\theta}_0^{(k)})]\frac{\partial \underline{\xi}_j}{\partial \underline{\theta}_0^{(k)}}, \tag{53}$$

y

$$\underline{R}_0^{(k)} = \sum_{t=1}^{N}\sum_{j=1}^{K} g_j(\underline{x}^{(t)}, \underline{\theta}_0^{(k)})[1 - g_j(\underline{x}^{(t)}, \underline{\theta}_0^{(k)})]\frac{\partial \underline{\xi}_j}{\partial \underline{\theta}_0}\frac{\partial \underline{\xi}_j}{\partial \underline{\theta}_0^T}. \tag{54}$$

En resumen, la actualización de los parámetros del modelo ME es el siguiente:

1. (Paso E) Calcular $h_j^{(k)}(t)$, $\psi_j^{(k)}(t)$, para cada experto ($j = 1,\ldots,K$), mediante las ecuaciones (24) y (43) respectivamente.
2. (Paso M) Estimar $\Sigma_j^{(k+1)}$, $\underline{\theta}_j^{(k+1)}$, $j = 1,\ldots,K$ y $\underline{\theta}_0^{(k+1)}$ mediante las ecuaciones (48) (49), (52).

6.3 Robustificación del Algoritmo de Aprendizaje basado en el Gradiente

Detalles de este algoritmo pueden ser revisados en [Tor03].

En particular, si se considera que las redes expertas y la red de agregación son redes lineales que siguen un modelo Gaussianoy si se utiliza la función softmax para las salidas de la red de agregación, dada por la ecuación (4), entonces la actualización de los parámetros de la red está dada por la siguiente expresión:

$$\begin{aligned}
\Delta\underline{\theta}_i &= \alpha\psi(\zeta)g_i\frac{1}{(2\pi)^{m/2}}\exp\left\{-\frac{(\underline{y}^{(t)}-\underline{\mu}_i)^T(\underline{y}^{(t)}-\underline{\mu}_i)}{2}\right\}(\underline{y}^{(t)} - \underline{\mu}_i^{(t)})\underline{x}^{(t)^T} \\
\Delta\underline{\theta}_{0i} &= \alpha g_i\left\{P(\underline{y}^{(t)}|\underline{x}^{(t)}, \underline{\theta}_i) - \textstyle\sum_k g_k P(\underline{y}^{(t)}|\underline{x}^{(t)}, \underline{\theta}_k)\right\}\underline{x}^{(t)^T}
\end{aligned} \tag{55}$$

donde α es la razón de aprendizaje.

7 Resultados y Comparaciones

El criterio de Prechelt [Pre94] establece que:

> Una evaluación a un algoritmo es denominada aceptable, si ésta utiliza un mínimo de dos problemas reales y compara esos resultados con al menos un método alternativo. Prechelt [Pre94].

Para mostrar el rendimiento que se obtiene al robustificar el algoritmo de aprendizaje basado en el gradiente y el algoritmo EM (REM-ME) se utilizan conjuntos de datos de regresión de dos fuentes: DELVE y PROBEN1.

DELVE (Datos para evaluar aprendizaje mediante experimentos válidos) es una colección de conjuntos de datos de muchas fuentes, un ambiente dentro de los cuáles los datos pueden ser usados para medir el rendimiento de los algoritmos de aprendizaje y un repositorio para los resultados de tales experimentos.

7.1 Experimentos para el Modelo ME en conjunto de datos *Boston* (DELVE)

El conjunto de datos 'Boston Housing Data' creada por David Harrison y Daniel Rubinfeld, está compuesta de 506 muestras, cada una con trece entradas, compuestas de 12 atributos continuos y un atributo binario representando las características de los vecindarios en el área de Boston y una salida, denominada 'atributo clase', representado el valor promedio de una casa en esos vecindarios. En [Tor03] se entrega una descripción y el rango de los atributos de entrada y salida, que son normalizados entre [-1,1] y luego desnormalizados para ser presentado en las tablas de resultados.

Para todos los experimentos que se realizaron en este trabajo, se dividió el conjunto de datos de tamaño N en 50% de los datos para el conjunto de entrenamiento, 25% de los datos para el conjunto de validación y el 25% restante de los datos para el conjunto de prueba. Nótese que el conjunto de datos original está ordenado según el vecindario. Los experimentos realizados fueron:

Debido a que el conjunto de Datos Boston original proviene de diferentes vecindarios de Boston donde las realidades económicas y culturales son distintas, es caracterizado como un problema con diferentes fuentes de datos apto para ser

modelado por un modelo ME. Las muestras tomadas para este problema son datos reales ruidosos. Por otro lado, la presentación de este conjunto de datos está ordenado por vecindario, lo que introduce una nueva dificultad en el sentido de que no todas las clases están igualmente representadas en los subconjuntos de datos (entrenamiento, validación, prueba) pudiendo incluso haber presencia en un subconjunto y ausencia en otro.

La conjetura realizada en este experimento es que la robustez en los algoritmos de aprendizaje permiten obtener mejores resultados en los conjuntos de prueba. El tamaño del conjunto de datos de entrenamiento es de 253 datos, lo cual nos entrega una cota superior para el número de expertos posibles de incluir en la mezcla, en este caso 15 expertos.

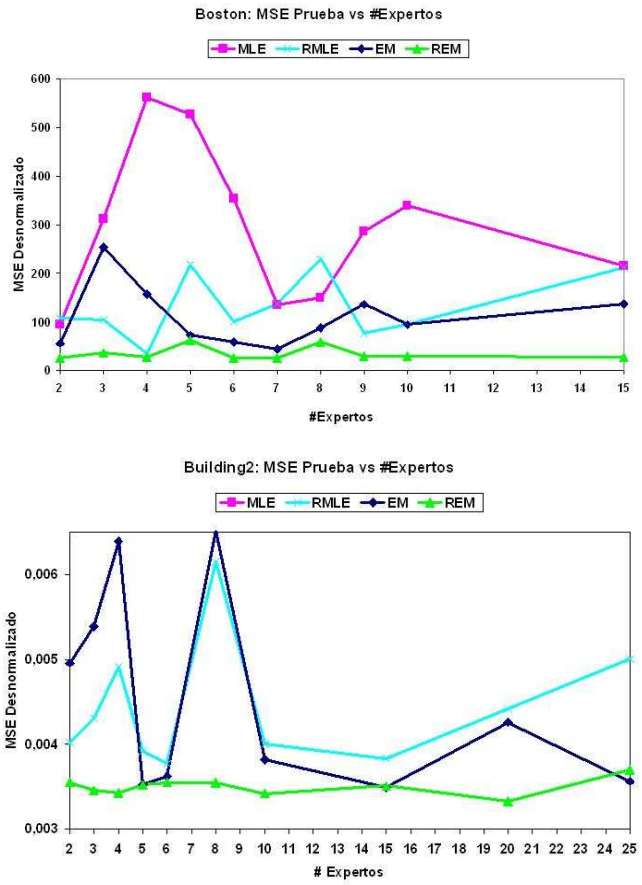


Figure 3: *Resultados*: MSE obtenido durante la fase de pruebas versus el Número de Expertos para el conjunto de datos Boston y el conjunto de Datos Building2 respectivamente. Notar que para el conjunto de datos Building 2, el gráfico es una ampliación de tres de los algoritmos, donde el algoritmo MLE por su bajo performance no aparece.

En la figura 3 se muestran los resultados obtenidos para el modelo de Mezcla de Expertos, utilizando los algoritmos basados en el gradiente, gradiente robusto, el algoritmo EM y REM.

En la Figura 4 se muestran los resultados obtenidos para el conjunto de datos con 2 expertos (ver [Tor03]).

Para este experimento, concluimos que el algoritmo REM presenta mejores resultados, debido a que es insensible a los datos atípicos, acotando la influencia que ellos ejercen en los parámetros del modelo ME. Aunque pareciera ser evidente que el algoritmo RMLE (gradiente robusto) presentase similares resultados, esta hipótesis se hace falsa debido a la incapacidad de este algoritmo de tomar ventaja de la estratificación de los datos de entrada, estratificación que se produce debido a que los datos provienen de distintos vecindarios de Boston, donde las realidades entre éstos no son iguales la mayoría de las veces. En cambio, el algoritmo REM se basa en el algoritmo EM, lo cual le adiciona la facultad de tomar ventaja de la modularidad del modelo, por lo que es más efectivo cuando los datos provienen de distintas fuentes.

Detalles acerca de otra serie de experimentos realizados sobre esta serie, tales como sobre el conjunto de datos ajustados

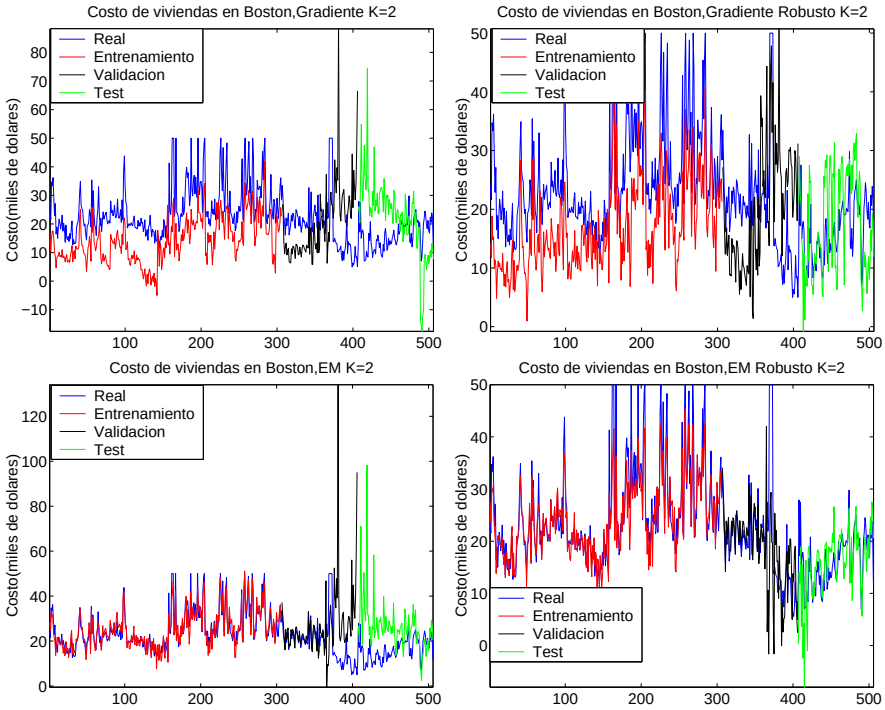


Figure 4: *Mezcla de* 2 *expertos*: Resultados para el conjunto de datos Boston Original utilizando 2 expertos y utilizando 4 algoritmos de aprendizaje diferentes. (arriba-izquierda) Gradiente, (arriba-derecha) Gradiente RObusto. (abajo-izquierda) Algoritmo EM. (abajo-derecha) Algoritmo REM.

o desordenados, donde también es mostrada la superioridad del algoritmo robusto pueden ser encontrados en [Tor03].

7.2 Experimentos para el Modelo ME en conjunto de datos *Building2* (PROBEN1)

Consiste de un problema de consumo de energía en un edificio. Este conjunto de datos *Building2* pertenece a la colección de problemas de aproximación para aprendizaje en Redes Neuronales, *PROBEN1* [Pre94]. Se trata de predecir el consumo de energía eléctrica, agua caliente y agua fría, basado en información acerca del día de la semana, hora del día, temperatura externa, humedad del aire externa, radiación solar y velocidad del viento. Por lo tanto el vector de entrada consiste de 14 atributos y el vector de salida es de dimensión 3. La muestra proviene de observaciones realizadas durante seis meses (desde Septiembre a Febrero). Acerca de los datos, observaciones de diversos investigadores, tales como [Pre94] observan que los requerimientos de agua caliente tienen una fuerte correlación con la temperatura externa [Tor03].

El tamaño del conjunto de datos, N es 4208 patrones, de los cuales 2104 patrones fueron utilizados para el conjunto de entrenamiento, 1052 patrones para el conjunto de validación y los 1052 datos restantes para el conjunto de prueba.

En la figura 3 se muestran los resultados obtenidos para el modelo de Mezcla de Expertos, utilizando los algoritmos basados en el gradiente, gradiente robusto, el algoritmo EM y REM.

De este experimento se puede inferir que robustificar el algoritmo de aprendizaje, algoritmos RMLE y REM, la complejidad es del modelo ME decrece en cuanto al número de expertos, y por tanto a la cantidad de parámetros a estimar del modelo. Esto es claro en la figura 3 donde se observa que para 3 o 10 expertos por ejemplo en el caso del algoritmo REM, se obtiene un error relativamente similar.

Según el test de Pretchel debemos comparar nuestros resultados con al menos un algoritmo alternativo. Para ésto. hemos entrenado una red de tres capas completamente conectada. El número de neuronas de la red en la capa escondida es escogido empíricamente, basándose en el rendimiento obtenido en el conjunto de prueba. Se utilizan distintas variantes de algoritmos de aprendizaje y optimizaciones para lograr un modelo de una única red de tres capas para compararlo con los resultados obtenidos por el modelo ME con algoritmos de aprendizaje MLE, RMLE, EM y REM.

Para el entrenamiento de la red multicapa se utilizan tres métodos alternativos para el aprendizaje:Algoritmo de aprendizaje backpropagation descendiente (GD), Algoritmo de aprendizaje backpropagation descendiente con momento (GDM) y Algoritmo de aprendizaje backpropagation basado en el método Levenberg-Marquardt(LM).

El resumen de los resultados pueden ser apreciados en la tabla 1 y 2.

Algoritmo	Entrenamiento	Validacion	Prueba
LM	2,260368	89,390206	50,9428
GDM	44,313272	84,339018	35,758352
MLE	44,945971	203,195314	134,599954
RMLE	52,781237	111,285037	34,943334
EM	9,208113	83,03982	44,594861
REM	12,152561	74,165716	25,197388

Table 1: *Boston Original*: Resumen de los resultados obtenidos sobre el conjunto de Datos Boston Original

Algoritmo	Entrenamiento	Validacion	Prueba
GD	0,007632	0,007555	0,007616
GDM	0,0076	0,0078	0,008
MLE	0,008324	0,008414	0,0086
RMLE	0,003832	0,004323	0,004
EM	0,003380	0,003704	0,003489
REM	0,003225	0,003574	0,003329

Table 2: *Building2*: Resumen de los resultados obtenidos sobre el conjunto de Datos Building2

En la Figura 5 se muestran los resultados obtenidos para el conjunto de datos con 33 expertos (ver [Tor03]).

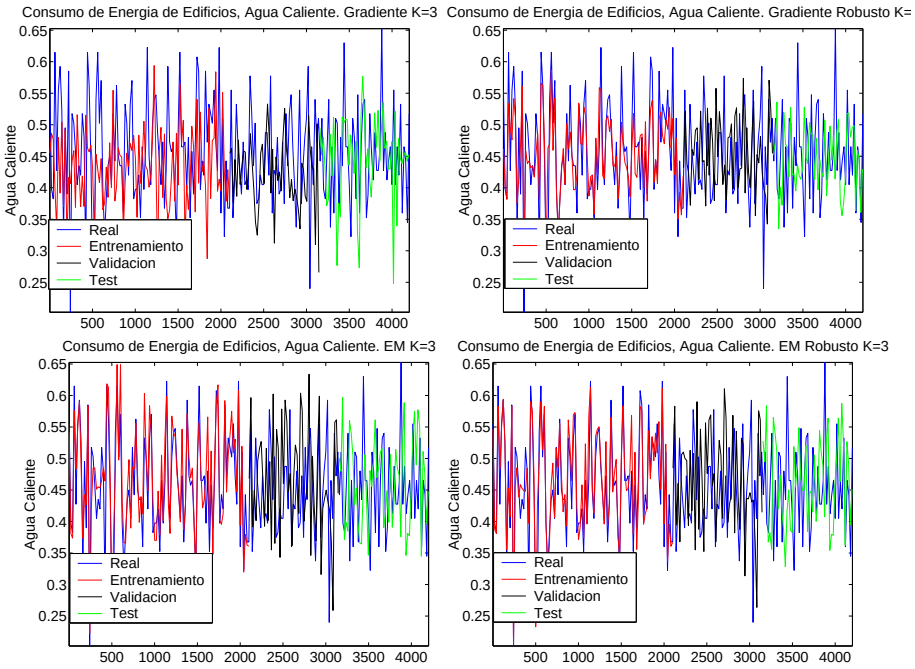


Figure 5: *Mezcla de* 3 *expertos*: Resultados para la predicción del consumo de agua caliente para el conjunto de datos Building 2 utilizando 3 expertos y utilizando 4 algoritmos de aprendizaje diferentes. (arriba-izquierda) Gradiente, (arriba-derecha) Gradiente RObusto. (abajo-izquierda) Algoritmo EM. (abajo-derecha) Algoritmo REM.

8 Conclusiones

Al utilizar el modelo Mezcla de Expertos con el algoritmo de aprendizaje robustificado en problemas de regresión (donde los datos presentaban datos atípicos) se lograron mejoras sinificativas según el test de Prechelt. Los conjuntos de datos reales que se utilizaron estaban altamente contaminados y al compararlo con las versiones no robustas, tanto el gradiente robusto y el REM mostraron ser superiores cuando aplicar robustez se justificaba, es decir cuando los datos contenían datos atípicos. La mejora que presenta este algoritmo es significativa según el test de Prechelt.

Basado en cada una de la hipótesis realizadas al comienzo de esta tesis, se hará un análisis de cada una.

Elección del Modelo ME: Basándonos en el conjunto de datos Boston original, donde los datos acerca del costo promedio de una casa dependía fuertemente de cuál vecindario se estaba interesado, lo cual podía ser visto como distintas fuentes de datos, se mostró que debido a la estratificación del espacio un modelo ME era el más adecuado, y en particular la utilización del algoritmo EM que saca ventaja de la modularidad del modelo obtenía un error cuadrático medio menor al presentado por otras técnicas alternativas.

Mejoras significativas en los conjuntos de prueba por utilizar algoritmos robustos durante el aprendizaje: esta hipótesis es verdadera en todos los casos de prueba siempre y cuando los datos presenten datos atípicos, es decir donde la robustez se justifica.

Reducción del número de expertos del modelo ME al utilizar algoritmos robustos durante el aprendizaje: esta hipótesis es verdadera. Nótese que cuando se utiliza algoritmos robustos de aprendizaje, el número de expertos necesarios para alcanzar igual precisión que las versiones de los algoritmos no robustos, es mucho menor, debido a que la robustez permite detectar aquellos datos que son atípicos y que por lo tanto no reflejan una real tendencia de los datos, entonces el modelo ME, no los trata como una nueva clase y por tanto no los modela de manera especial asignándoles un experto o una mezcla de expertos nueva sino que los agrupa a una clase de datos similar, que es donde realmente corresponde.

El modelo ME converge al menos un orden de magnitud más rápido si se utiliza durante el aprendizaje el algoritmo EM en vez de un algoritmo basado en el gradiente: esta hipótesis fue confirmada debido a que el algoritmo EM saca ventaja de la modularidad del modelo ME, y por lo tanto la validez de esta hipótesis es también para datos que provienen de distintas fuentes.

Cuando los conjuntos de datos presentan datos atípicos, el modelo ME con aprendizaje robusto presentará mejoras significativas en el rendimiento: siempre que los conjuntos de datos presenten datos atípicos o contaminados, las versiones robustas de los algoritmos de aprendizaje son superiores no sólo en las etapas de entrenamiento y validación sino en la de prueba, siendo esta más importante debido a que corresponde a un conjunto de datos antes no visto por el sistema.

Gracias a los experimentos realizados en esta tesis, podemos concluir que, el rendimiento del modelo Mezcla de Expertos presentado en [JJNH91b], puede ser mejorado si se utilizan algoritmos robustos de aprendizaje. En particular si se utiliza el algoritmo REM, se obtiene un modelo ME, de rápida convergencia, insensible a los datos atípicos en el sentido que extrae la información relevante de éstos pero acotando su impacto, de alta capacidad de generalización durante la fase de prueba o funcionamiento del sistema, sin aumentar la complejidad de la arquitectura, pues puede trabajar con el número mínimo de expertos necesarios sin degradar su rendimiento. Si se utiliza el algoritmo robusto basado en el gradiente, referido en esta tesis como RMLE, se obtienen las mismas bondades que para el algoritmo REM, a excepión de la rápidez de convergencia debido a que los algoritmos basados en el gradiente, tanto robustos como no robustos, no sacan ventaja de la modularidad del modelo de mezcla.

References

[ABH+72] D. Andrews, P. Bickel, F. Hampel, P-Huber, W. Rogers, and J. Tukey. Robust estimate of location: Survey and advances. Technical report, Princeton University Press, Princeton, N.J., 1972.

[AMS01] H. Allende, C. Moraga, and R. Salas. Neural model identification using local robustness analysis. *Lecture Notes in Computer Science. Fuzzy Days 2001*, 2206:162–173, Nov 2001.

[AMS02] H. Allende, C. Moraga, and R. Salas. Robust estimator for the learning process in neural networks applied in time series. *Lecture Notes in Computer Science, ICANN 2002*, 2415:1080–1086, Aug 2002.

[AMST04] H. Allende, C. Moraga, R. Salas, and R. Torres. Modular Neural Network applied to non-stationary Times Series. *Accepted in Advances in Soft Computing*, 2004.

[ATSM03] H. Allende, R. Torres, R. Salas, and C. Moraga. Robust learning algorithm for the mixture of experts. *Lectures Notes in Computer Science. IBPRIA2003*, 2652:19–27, Jun 2003.

[Cam84] N. A. Campbell. Mixture models and atypical values. *Math. Geol.*, pages 465–477, 1984.

[CM94] J.T. Connor and R.D. Martin. Recurrent neural networks and robust time series prediction. *IEEE Transactions of Neural Networks*, 2(5):240–253, 1994.

[DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. 39:1–38, June 1977.

[HRRS86] F.R. Hampel, E.M. Ronchetti, P.J. Rousseeuw, and W. A. Stahel. *Robust Statistics, The approach based on Influence Functions*. Wiley Series in probability and mathematical statistics, 1986.

[Hub64] P. J. Huber. Robust estimation of a location parameter. *Ann. Math. Statist.*, 16, 1964.

[JJ91] R. A. Jacobs and M. I. Jordan. A modular connectionist architecture for learning piecewise control strategies. *Proceedings of the American Control Conference*, 2.:1597–1602, 1991.

[JJ92] M. I. Jordan and R. A. Jacobs. Hierarchies of adaptive experts. pages 985–992. 1992.

[JJ94] M.I. Jordan and R.A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6(2):181–214, 1994.

[JJ99] M. Jordan and R. Jacobs. *Modular and hierarchical learning systems, The Handbook of Brain Theory and Neural Networks, Cambridge, MA*, volume 1. MIT Press, 1999.

[JJB91] R. A. Jacobs, M. I. Jordan, and A. G. Barto. Task decomposition through competition in a modular connectionist architecture - the What and Where vision tasks. *Cognitive Science*, 15(2):219–250, 1991.

[JJNH91a] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

[JJNH91b] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

[Mar76] R. A. Maronna. Robust M-estimators of multivariate location and scatter. *Ann. Statist.*, 4, 1976.

[MP01] G. McLachlan and T. David Peel. *Finite Mixture Models*, volume 1. Wiley Series in Probability and Statistics, 2001.

[Pre94] L. Prechelt. PROBEN1 – a set of benchmarks and benchmarking rules for neural training algorithms. *Technical Report 21/94, Fakultat fur Informatik, Universitat Karlsruhe, D-76128 Karlsruhe, Germany*, 1994.

[Tor03] R. Torres. Algoritmo Robusto de Aprendizaje para el Modelo Mezcla de Expertos. *Tesis de Magíster en Ciencias de la Ingeniería Informática, Universidad Técnica Federico Santa María*, Noviembre, 2003.

[TSAM02] R. Torres, R. Salas, H. Allende, and C. Moraga. Estimador robusto en modelos de mezcla de expertos locales. *CLATSEV*, Nov 2002.

[TSAM03] R. Torres, R. Salas, H. Allende, and C. Moraga. Robust expectation maximization learning algorithm for mixture of experts. *Lectures Notes in Computer Science. IWANN2003*, 2686:238–245, Jun 2003.

Resumenes/Abstracts

Resumen: Clustering Very Short Documents based on Grouping Keywords

Mikhail Alexandrov, Alexander Gelbukh, Paolo Rosso

e-mail: dyner1950@mail.ru, www.gelbukh.com, prosso@dsic.upv.es

National Polytechnic Institute – México
Polytechnic University of Valencia – España

Abstract

Introduction. By very short documents we mean first of all abstracts of no more than 50-100 words, which constitute the major part of the contents of freely accessible digital libraries on the Internet. The main problem in clustering such documents consists in very low absolute frequency of keywords (1-3), which leads to unstable results. In this paper we elaborate an approach based on grouping keywords to compensate for their sparseness in the texts.

Algorithms. We present two simple algorithms for document clustering, both relying on clustering keywords. The first one clusters documents considering every group of keywords a new coordinate in the index space, equal to the sum of keyword occurrences of a given group. The second one distributes documents between clusters considering every group of keywords as a cluster description and calculating their contribution to a given document. Both algorithms critically depend on the number of keyword clusters, which is chosen by the user or defined by the classical geometric method.

Experiments. The experiments were carried out with the abstracts of two conferences downloaded from Internet and marked up by experts, of about one hundred documents. We applied our algorithms to different keyword lists, with different numbers of keyword clusters and document clusters. For clustering we used the cosine measure with K-medoid and NN-methods. For document distribution we used the linear and Euclidean measure. Generally, grouping keywords led to 5-10 stable clusters when initially we had one or two hundreds of keywords. The results were evaluated by cluster coincidence with the manually marked up corpus.

Conclusions. Our experiments have shown 10 %-20 % improvement over the traditional tf-idf techniques used for clustering full-text (long) documents. The suggested approach can be used when the keywords are strongly enough interdependent (otherwise, the resulting clusters are too fuzzy) and for large enough document sets (otherwise, the clusters prove to be imprecise and unstable). In the future we plan to check the algorithms on larger document sets and modify them to combine statistical similarity measures with an ontology.

Keywords:Short documents, Document clustering, Clustering

Resumen: Representing Clusters by Typical Documents for Navigating the Search Results in Relevance Feedback Procedure

Ales Bourek, Mikhail Alexandrov, Alexander Gelbukh

e-mail: bourek@med.muni.cz, dyner1950@mail.ru, www.gelbukh.com

Masaryk University in Brno – República Checa
National Polytechnic Institute – México
Polytechnic University of Valencia – España

Abstract

Introduction. Relevance feedback is a well-known technique for interactive improving of the user query in information retrieval. First, the user formulates the query to the search engine. Then in iterative manner (1) the user inspects the search results marking some documents as relevant or irrelevant for his or her information need, and (2) basing on this information, the system corrects the user query to better correspond to his or her information need, executes the corrected query, and presents the new results to the user, after which the process repeats. The process is finished when the user is satisfied with the results or runs out of time. Usually, the documents are presented to the user in order of relevance. This leads to the following shortcoming: the user examines a great number of similar documents located near the top of the list, but never gets to significantly different documents located far from the top of the list, which limits the information received by the system for re-calculating the query. To solve this problem, we propose to present the documents to the user in clusters, to facilitate the exploration of different parts of the collection. We build on our previous work on selection of typical documents in clusters.

Method. The only difference of our method as compared with the relevance feedback procedure described above is that the documents are presented to the user in clusters rather than in a plain list. Here the usual technics of document clustering in the index space is used. The clusters presented to the user are ordered by the average relevance of a document in the cluster. The user is asked to mark whole clusters as relevant or irrelevant. With this, a greater number and a wider variety of the documents will receive some evaluation (positive or negative) than with the usual procedure when the user only inspects a small number of similar documents located at the top of the list. The clusters are presented to the user by their typical documents: (a) the average (central) document of a given cluster reflecting its main idea, (b) the closest one to other documents in the collection, and (c) the least similar one to all other documents in the collection. Usually the user can judge on the relevance of the whole ccluster by only the central representative document. However, if this information is not enough to make a clear decision, the user has to look through the other representative documents, which in a sense delimit the clusters from the opposite sides. Because the relevance feedback procedure changes the query, the contents of clusters also change on every step of the search.

Experiments. For our experiments we used 500 English abstracts of medical articles from the e-library of the Czech Center for Quality in Healthcare, Prague. We tried index lists of different sizes and experimented with different number of clusters. The results were evaluated by calculating the number of documents inspected by the user until a given number of relevant documents is found. The experiments show that with the appropriate number of clusters the user needs to manually inspect a smaller number of documents

Conclusion. We have suggested a method for navigation in the search results returned by a search engine in the relevance feedback cycle. The method consists in presenting the documents to the user in clusters rather than as a plain list. To present a cluster to the user, a typical document and the documents delimiting the cluster from the opposite sides are used. With this method, the user has to inspect a smaller number of documents, while the system receives richer information for the relevance feedback re-weighting. While this is an ongoing project, the results obtained so far are promising.
Keywords:Information retrieval, Relevance feedback, Document clustering, Typical document

Lista de Autores

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

V

W

X

Y

Z

Artículos por país

Alemania

Argentina

Brasil

Chile

Colombia

Costa Rica

España

Estados Unidos

Francia

Holanda

Italia

México

Paraguay

Perú

Portugal

República Checa

Sudáfrica

Túnez

Uruguay

Venezuela

Organizadores

SPC
SOCIEDAD PERUANA DE COMPUTACIÓN

CLEI
CENTRO LATINOAMERICANO DE ESTUDIOS EN INFORMÁTICA

Apoyo

Sociedad Chilena de Ciencia de la Computación

USP ICMC

GUPAC int.

Auspiciadores

IFIP
The International Federation for Information Processing

IEEE COMPUTER SOCIETY

UNESCO

Universidad Católica San Pablo

Universidad Nacional de San Agustín

Universidad Católica Santa María

UNIVERSIDAD TECNOLOGICA DEL PERU

CLINICA ODONTOLÓGICA ODAM

INCAMACH
Soluciones Informáticas
www.incamach.tk

NCA CONSULTORES
Investigación • Desarrollo • Innovación
nca_consultores@msn.com
Arequipa - Perú

VisionLINUX
Soluciones Profesionales basadas en Linux

CONCYTEC