MDA View of Enterprise Architecture Frameworks

(Una Visión MDA de Frameworks de Arquitectura Empresarial)

Francisca Losavio
Universidad Central de Venezuela, Centro ISYS, LaTecS, Facultad de Ciencias,
Caracas, Venezuela.
flosav@cantv.net

and

Dinarle Ortega
Universidad de Carabobo, Departamento de Computación,
Valencia, Venezuela.
dortega@uc.edu.ve

and

María Pérez
Universidad Simón Bolívar, Departamento de Procesos y Sistemas,
Caracas, Venezuela.
movalles@usb.ve

Abstract

Model Driven Architecture (MDA) represents an effort towards standardization and integration of software development models. Different domains are now experimenting with MDA, in particular the Enterprise Application Integration (EAI) defined as the process of integrating heterogeneous software applications. Enterprise Architecture (EA) frameworks consider in general also the modeling of integration. In particular, the Enterprise Application Integration Framework (EAIF), developed by our research group, is an enterprise architecture framework which provides sound and unified definitions of the modeling elements involved in the EAI domain. The main goal of this work is to establish a mapping of the MDA models to the EAIF abstraction levels. Moreover, this correspondence is also defined for the elements of known EA frameworks that can be used for EAI modeling. The approach is illustrated with a Customer Relationship Management (CRM) case study. This work can help to improve the standardization efforts in the context of modeling engineering, facilitating the development of business applications with standard modeling techniques.
Keywords: Enterprise Architecture Frameworks, MDA, EAI, EAIF.

Resumen

La Arquitectura Dirigida por Modelos (del inglés, Model Driven Architecture, MDA) representa un esfuerzo hacia la estandardización y la integración de los modelos del desarrollo de software. En la actualidad, diversos dominios están experimentando con MDA, en particular la integración de Aplicaciones Empresariales (del inglés, Enterprise Application Integration- EAI) definida como el proceso de integrar aplicaciones de software heterogéneas. Los frameworks para la arquitectura empresarial se pueden utilizar para modelar la integración. En particular, el Enterprise Application Integration Framework (EAIF), desarrollado por nuestro grupo de investigación, es un framework de arquitectura empresarial, el cual proporciona definiciones unificadas y consistentes de los elementos de modelación involucrados en el dominio de la EAI. El objetivo principal de este trabajo es establecer una correspondencia de los modelos de MDA con los niveles de abstracción de EAIF. Además, la misma correspondencia se define para los elementos de frameworks de arquitectura empresarial conocidos, que pueden ser utilizados para modelar la EAI. El enfoque se ilustra con un estudio de caso de un sistema de Gestión de Relación del Cliente (del inglés, Customer Relationship Management-CRM). Este trabajo puede ayudar a mejorar los esfuerzos de estandardización en el contexto de la ingeniería de modelos, facilitando el desarrollo de aplicaciones de negocio con técnicas de modelación estándar.
Palabras claves: Frameworks de Arquitectura Empresarial, MDA, EAI, EAIF.

The OPSU of Venezuela Alma Mater program has provided financial support for the publication and presentation of this work at CLEI'06, Chile

1. INTRODUCTION

Software integration is the "glue" allowing the smooth flow of information between the different organization levels. This information must be shared within the organization to guarantee a better planning, control and evaluation of these processes, inside and outside of the organization. However, the software integration problem has not been entirely solved and it is still a very expensive process in terms of human and technological efforts [26]. The Enterprise Application Integration (EAI) approach was conceived as a solution to this problem. EAI is defined as the process of coordinating and synchronizing multiple heterogeneous software applications. One of the most crucial points of handling application integration is to combine all the organization assets or expertise, including information technology such as data base technology, distributed and real-time capabilities and middleware components [19] to support the complexity of the business processes which are related to the services provided by the organization. It is crucial to count with powerful tools for decision making in integration projects. Several Enterprise Architecture (EA) frameworks have been proposed up to now to model different aspects of an integration problem (business processes, applications) [5], [25], [28], [29] and [34]. In this context, the term framework is considered here as a set of assumptions, concepts, values, and practices that constitutes a way of viewing reality [25] and EA means a strategic information asset base, which defines the business in terms of the information necessary to operate the business (processes and services) and the technologies necessary to support the business operations [4]. In particular the EAIF (Enterprise Application Integration Framework) [17] is an EA framework, integration oriented, to specify integrated business processes, enterprise systems and middleware. It is expressed in UML [23], and it is independent of specifics methodologies and technologies. It can be used as a tool to document and identify precisely central aspects of integration platforms. In particular, aspects that are missing or need improvement can be detected.

On one hand, the Object Management Group's Model Driven Architecture (MDA) [22] is a generic solution to the integration problems using reusable platform independent models, transforming them into platform specific models, to facilitate software integration and to target automatic code generation, ensuring consistency among the models used. It represents a major effort to create the standards necessary to facilitate a comprehensive new approach to the creation, integration, and maintenance of software assets. It separates the fundamental logic behind a specification from the specifics of the particular middleware that implements it. MDA addresses the challenges of today's highly networked, constantly changing systems environment, providing an architecture that is claimed to assure portability, interoperability, domain specificity and productivity [6]. However, since it is an emerging discipline, it still lacks precise guidelines, mature methods and tools. The MDA approach is concerned with organizing the models used in the software development process so that developers can move from abstract models to more concrete models. This emphasizes the use of Computation Independent Models (CIMs), Platform Independent Models (PIMs), Platform Specific Models (PSMs) and mappings that allow a developer to transform one model into another. A mapping may be expressed as associations, constraints, rules, templates with parameters that must be assigned during the mapping [10]. A possible taxonomy of several existing and proposed model transformation approaches is presented in [13].

It is important to point out that MDA is an approach that can be used with existing methodologies [7] and it is primarily concerned with providing freedom in the selection of middleware technology.

Then we can map elements of enterprise architecture frameworks to the different types of MDA models: CIM, PIM, PSM. The main goal of this work is to describe the mapping of these MDA models to EAIF. It also introduces several other well know EA frameworks; in particular they can be used for EAI modeling and explains how they also map to the MDA models. The transformation rules between the MDA models are given for EAIF. Finally, the EAIF models are applied to an e-business application using J2EE technology, focusing on integration aspects. The case study concerns a Customer Relationship Management (CRM) application, which is a widely used business strategy involving the customer and its relation with the organization.

This paper is structured as follows, besides this introduction and the conclusions: Section 2 presents EAIF and its MDA mapping and the related works, the known EA frameworks: Zachman, Stojanovic and Cummins, including their mapping. Finally, section 3 presents the EAIF models applied to a CRM case study.

2. MAPPING MDA MODELS TO EA FRAMEWORKS

This section presents EAIF and its MDA mapping and the related works, the known EA frameworks: Zachman, Stojanovic and Cummins, including their mapping-

2.1 The EAI Framework (EAIF)

EAIF models the EA, focusing on EAI aspects. It provides organized and unified definitions of the elements related with people, processes, services and mechanisms within the enterprise [15], [16] and [17]. EAIF extends the BCMI [2] which proposes a three-tier framework for an integrated CASE environment: *processes*, *services* and the *mechanisms* level; and *implementation* and *adaptation* relations. On the other hand, EAIF considers also the Sandoe's EAI views [26]: *backward*, *forward* and *upward*. The EAIF UML [23] class diagram [17] is a conceptual model of the notions used in the EAI. It considers the following levels (see figure 1):

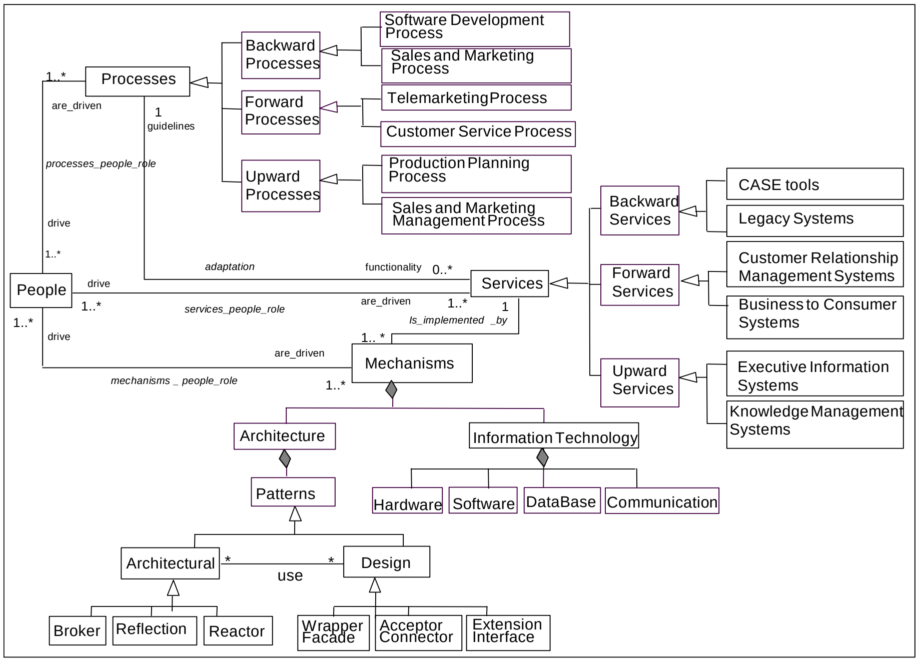


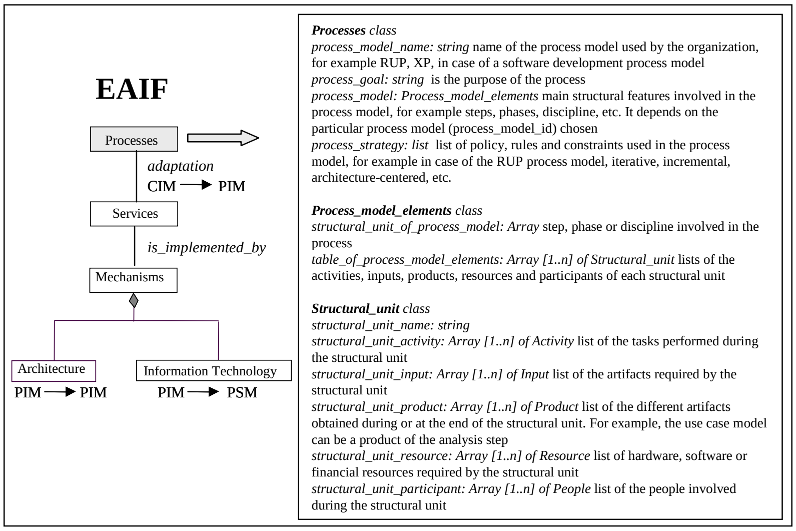
Figure 1. EAIF expressed as an UML model [17]

1. *Processes*. Business processes [34] are described for each integration view. Notice that each business process has its own goals and constraints and it depends directly from the specific functionality that is required by the integrated system.
2. *Services*. They are presented in terms of the typical IS supporting the processes according to the three integration views [15].
3. *Mechanisms*. It is independent from the views and is considered the EAI middleware [15] and [20]. Two types of components are considered.
   a) The *Architecture* (components and connectors with a behavior [27]) is the main mechanism to articulate the services supporting the business processes, in particular the integration processes. Only the main architectural styles and design patterns [3] considered for the EAI middleware are included in Figure 1.
   b) The *Information Technology* (IT) is considered a set of resources available to manage changes and to give support to people developing their activities [14], [21] and [26].

In order to reduce complexity, the figure 1 shows only the class, relationship and role names are shown. The navigability of the associations is bidirectional and only a limited range of very well known processes, services and mechanisms is shown in the class diagram. It is important to point out that only some examples of types of processes, services and mechanisms that can be used in EAI are shown.

Part of the textual specification of the EAIF main classes: Processes, Services and Mechanisms [18] can be seen in the figures 2, 3 and 4, respectively, where the gray arrow indicates the level that is being specified. The textual specification has a “natural language” flavor to increase legibility; common types such as “string”, “array” and “list” are used. Only the class attributes are shown (in italics) with the respective comments (no italics). The specification of the People level is not shown, to abridge this presentation.

Note that in the *Functionality* class (see figure 3), the functionality or service is identified by its name and the list of the qualities it should provide. In this list, only the name of the quality characteristics is given, like “reliability” or “efficiency” and eventually some sub-characteristics. However, according to the ISO 9126-1 standard [12], the specification could be refined including details for the attributes or measurable items, such as metrics and values assigned to each quality.

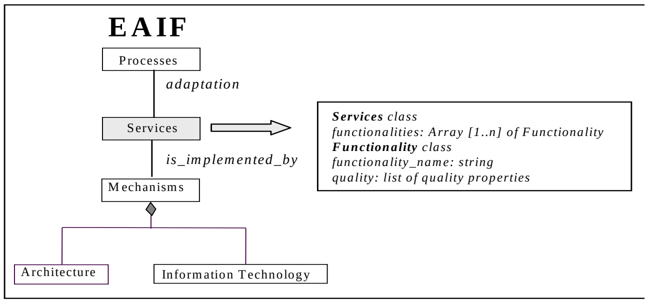


***Processes** class*
process_model_name: string name of the process model used by the organization, for example RUP, XP, in case of a software development process model
process_goal: string is the purpose of the process
process_model: Process_model_elements main structural features involved in the process model, for example steps, phases, discipline, etc. It depends on the particular process model (process_model_id) chosen
process_strategy: list list of policy, rules and constraints used in the process model, for example in case of the RUP process model, iterative, incremental, architecture-centered, etc.

***Process_model_elements** class*
structural_unit_of_process_model: Array step, phase or discipline involved in the process
table_of_process_model_elements: Array [1..n] of Structural_unit lists of the activities, inputs, products, resources and participants of each structural unit

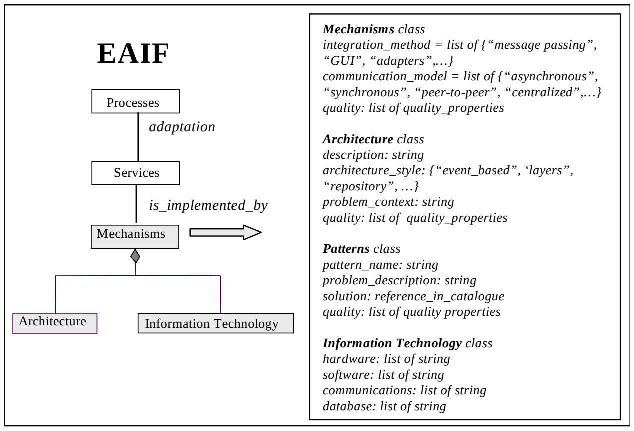
***Structural_unit** class*
structural_unit_name: string
structural_unit_activity: Array [1..n] of Activity list of the tasks performed during the structural unit
structural_unit_input: Array [1..n] of Input list of the artifacts required by the structural unit
structural_unit_product: Array [1..n] of Product list of the different artifacts obtained during or at the end of the structural unit. For example, the use case model can be a product of the analysis step
structural_unit_resource: Array [1..n] of Resource list of hardware, software or financial resources required by the structural unit
structural_unit_participant: Array [1..n] of People list of the people involved during the structural unit

Figure 2. Specification of the Processes level



***Services** class*
functionalities: Array [1..n] of Functionality
***Functionality** class*
functionality_name: string
quality: list of quality properties

Figure 3. Specification of the Services level



***Mechanisms** class*
integration_method = list of {“message passing”, “GUI”, “adapters”,…}
communication_model = list of {“asynchronous”, “synchronous”, “peer-to-peer”, “centralized”,…}
quality: list of quality_properties

***Architecture** class*
description: string
architecture_style: {“event_based”, ‘layers”, “repository”, …}
problem_context: string
quality: list of quality_properties

***Patterns** class*
pattern_name: string
problem_description: string
solution: reference_in_catalogue
quality: list of quality properties

***Information Technology** class*
hardware: list of string
software: list of string
communications: list of string
database: list of string

Figure 4. Specification of the Mechanisms level

Figure 5 shows the specification of the *adaptation* and *is_implemented_by* associations (see Figure 1).

***Adaptation** association*	***is_implemented_by** association*
association end 1 = Services	*association end 1 = Services*
association end 2 = Processes	*association end 2 = Mechanisms*
association name = “adaptation”	*association name = “is_implemented_by”*
type = “association”	*type = “association”*
role 1 = “functionality”	*role 1 = “unspecified”*
role 2 = “guidelines”	*role 2 = “unspecified”*
*multiplicity 1 = 0..**	*multiplicity 1 = 1*
multiplicity 2 = 1	*multiplicity 2 = 1..**

Figure 5.Specification of the *adaptatio*n and *is_implemented_by* associations

- MDA & EAIF

In what follows, the justification of the mappings between the EAIF levels and the MDA models is presented. The CIM includes the people and processes levels. In the Services level the analysis model of the enterprise systems is modeled by the PIM. With respect to the Mechanisms level, the architecture is also part of the PIM, since it is a high abstraction level specification, such as an ADL. The technology is modeled by PSM. Notice that the Services level is independent from technological changes.

Transformation from *Processes to Services* levels (CIM ->PIM)
A *services* class according to the *adaptation* relationship (see Figure 1, 2, 3, 5 and 7) adapts one *processes* class. Notice that there may be services that are not adapted to any process. In this context "to adapt" means that, according to the roles of the *adaptation* association, a service corresponds to a system's functionality and a functionality of a service has guidelines describing the selection of the corresponding service.

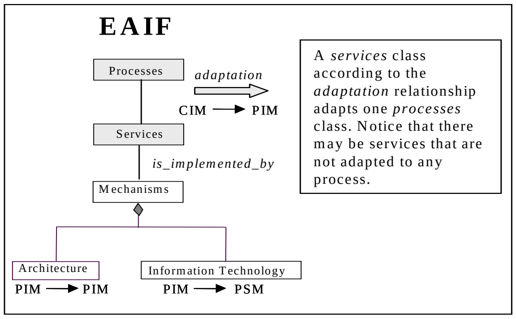

EAIF
Processes
adaptation
CIM → PIM
Services
is_implemented_by
Mechanisms
Architecture
PIM → PIM
Information Technology
PIM → PSM
A *services* class according to the *adaptation* relationship adapts one *processes* class. Notice that there may be services that are not adapted to any process.

Figure 7. Transformation from processes to services levels

Transformation from *Services to Mechanisms* levels
A *mechanisms* class according to the *is_implemented_by* relationship (see Figure 1, 3, 4, 5 and 8) implements one *services* class. Notice that *Architecture* (PIM->PIM) and *Information Technology* (PIM-> PSM) classes are part of the *Mechanisms* class and they must be specified accordingly.

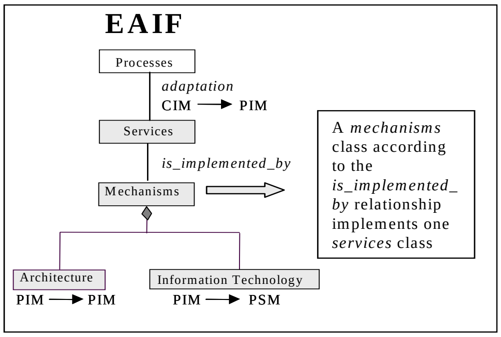


Figure 8. Transformation from services to mechanisms levels

2.2 Others EA frameworks
This section presents briefly some widely accepted and mature EA frameworks that can be used to model EAI. They have been considered to define EAIF in [17]. Their elements are mapped to the MDA models justifying each choice.

2.2.1 Zachman's Framework.
It conceptualizes a specific architecture of an organization, integrated into a unique scheme [25] and [33]. It consists of a classification scheme that organizes descriptive representations in two dimensions. The vertical dimension describes the perspectives of the stakeholders (Planner, Owner, Designer, Builder and Sub-contractor). The last row represents the "real world", the actual running elements of the organization. The horizontal dimension describes the types of abstractions that define each perspective. These abstractions are based on widely used questions that people usually ask to understand. They are Data, Process, Network, People, Time and Motivation. This framework can be used to analyze any business object or enterprise portion. It does not propose any methodology to build EAs. The Open Group Architecture Framework (TOGAF) [24], inspired in the Zachman's framework proposes a specific methodology to construct EA.

- MDA & the Zachman's Framework

Frankel et al. in [9] presents a mapping between MDA and this framework (see Table 1).

Table 1. The MDA models mapping to Zachman

Zachman perspectives	Zachman abstractions	MDA models
BUSINESS MODEL (Conceptual) ***Owner***	- Semantic Model (Data) - Business Process Model (Process) - Business Logistics System (Network) - Work Flow Model (People) - Master Schedule (Time) - Business Plan (Motivation)	CIM, depending on the exact content of the Contextual Row, it is suggested that discrete items, such as lists or goals should not be considered by MDA
SYSTEM MODEL (Logical) ***Designer***	- Logical Data Model (Data) - Application Architecture (Process) - Distributed System Architecture (Network) - Human Interface Architecture (People) - Processing Structure (Time) - Business Rule Model (Motivation)	PIM, logical designers create Platform Independent Models (PIM) to illustrate the organization's architecture, without reference to any specific implementation
TECHNOLOGY MODEL (Physical) Builder	- Physical Data Model (Data) - System Design (Process) - Technology Architecture (Network) - Presentation Architecture (People) - Control Structure (Time) - Rule Design (Motivation)	PSM developers and testers will use tools to generate code and test cases from the PIM architecture

2.2.2 Stojanovic's Framework.

It defines an integrated, effective, and flexible approach consolidated in a component-based framework providing comprehensive concepts, models, rules, methods and guidelines as a support for advanced enterprise systems development [28] and [29]. It provides a rich specification approach for defining not only behavioral and structural aspects of complex enterprise systems, but also aspects including a significant human and organizational component. It offers an integrated view on the system through various viewpoints which evolve coordinately through time, using a consistent component-based way of thinking. It is based on the Reference Model of the Open Distributed Processing (RM-ODP) [11] standardization efforts and best practices and the Component-Based Development (CBD) approach [30]. The models involved are Enterprise Architecture Model, System Architecture Model and Distribution Architecture Model. The aim is to provide a complete system specification, from concept to deployment, in a rigorous and consistent way.

- MDA & the Stojanovic´s Framework

Table 2 presents the MDA models mapping to the Stojanovic levels. In [8] a similar mapping is presented for RM-ODP.

Table 2. The MDA models mapping to Stojanovic

Stojanovic levels		**MDA models**
Enterprise Architecture Model/ Enterprise viewpoint		CIM. It specifies the behavior of the system in the business context. It focuses on the purpose, scope and policies governing the activities of the specified enterprise system. It includes the marketing concerns relevant to defining a product line
System Architecture Model	Information viewpoint	CIM. It presents an abstract view of the information. It focuses on the kinds and semantics of the information handled by the system, as well as information processing and constraints on it
	Computational view point	PIM. It provides the business services related with the business requirements. It is used to specify the functionality of an ODP system, with transparent distribution facility [28]
Distribution Architecture Model/Engineering viewpoint		PIM. It specifies a distributed infrastructure in terms of its distribution services, location and communication in the n-tiers architecture

Stojanovic Framework is related with the abstract model of the RM-ODP and is independent of technological aspects; the PIM can be mapped to a PSM to target platforms like the CORBA Component Model, Enterprise Java Beans or Microsoft COM+.

2.2.3 Cummins's Framework.

It concentrates on the EAI implementation aspects, detailing business management and infrastructure aspects [5]. It depicts enterprise integration from a management viewpoint in four different domains: users, business process, applications and infrastructure. It focuses on the notion of Enterprise Integration Architecture (EIA) to establish a set of characteristics that the enterprise must posses to perform software integration: distributed computing, component-based applications, event-driven processes, loose coupling of business functions, decision support information, workflow management, internet access and personalization of interfaces. The goal of EIA is to manage the business process with workflow management facilities so that the processes are visible and manageable.

- MDA & the Cummins´ Framework

Table 3 suggests how the MDA models might map to the Cummins levels.

Table 3. How the MDA models map to the Cummins Framework

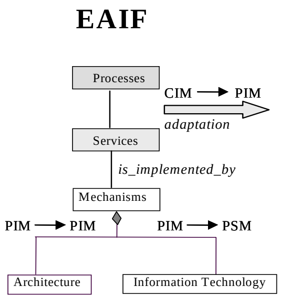
Cummins management domains	**MDA models**
Users	CIM, enterprise integration requires changes in the roles of people and the way they think about and do their work. People must learn to work with the processes, and they must have the opportunity to tailor or improve the processes, within limits.
Business Processes	
Applications	PIM, three application types are considered: legacy systems, component-based applications to construct and adapt applications by incorporating replaceable, shareable building blocks and business applications to provide a set of closely related business functions.
Infrastructure	PSM, computers, networks, software and associated services that support the operation and interconnection of many systems

3. E-business Application: a CRM Case Study

The *Customer Relationship Management* (CRM) approach is a widely used and still evolving business strategy. It is centered on the customer and its relation with the organization [32]. CRM addresses all the customer touch points, such as face-to-face, Internet, or phone [26]. It integrates sales, marketing and service strategies. It helps to establish collaborative relationships with customers on a long-term basis, using information technology as such as, databases, data warehouses, and data mining.

J-énesis is a CRM application for the *customer retention and loyalty phase* [31]. It offers Internet registration services for different software courses. Products and services consider the customer's needs (profiles) [1]. Java 2 Platform Enterprise Edition (J2EE) [30] is used as the integration mechanism. The users of J-énesis are: customers, customer service analysts and marketing analysts and it does not support a call center service. After instantiating EAIF with the J-énesis case study, Figures 9, 10 and 11 show the models transformation CIM to PIM, PIM to PIM and PIM to PSM, respectively.

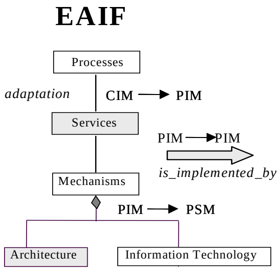
Figure 9 describes the transformation from the Processes level (CIM) to the Services level (PIM), which is characterized by the *adaptation* relationship. For example, the functionalities *collect*, *recall and update software courses* and *collect, recall and update customer retention strategies* of *J-énesis_service01* adapts the activity *recall software courses and retention and loyalty strategies data* of the *J-énesis_forward_process*. Figure 10 shows the transformation from the Services level (PIM) to the Mechanisms/architecture level (PIM), which is characterized by the *is_implemented_by* association. In this case, we have a layered architecture, where the design solutions (*pattern01*, *pattern 02*) implement the functionalities of *J-énesis_service01*. Finally, figure 11 shows the transformation from the services level (PIM) to the mechanisms/information technology level (PSM), which is characterized by the *is_implemented_by* association.



```
J-énesis_forward_process class extends Customer Service
process_model_ name = “based in CRM ”
process_goal: “this process administers the information related with software courses, customer profile and
customer retention and loyalty strategies. Also, it allows the access to remote resources to customer”
process_model: Process_model_elements02
process_strategy = “increase customer retention and loyalty using incentives and allow access to relevant
information on client and courses through Internet”
Process_model_elements02 class instantiates Process_model_elements
structural_unit_of_process_model: Array of structural_unit_of_process_model01[1]=“step”
table_of_process_model_elements: Array_of_structural_unit
where
                              …
table_of_process_model_elements [3] =
(structural_unit_name = “recall software courses and retention and loyalty strategies data”
structural_unit_activity: Array [1..2] of Activity where
  structural_unit_activity [1] = (“recall software courses data”,…)
                              …
structural_unit_input: Array [1..2] of Input where
  structural_unit_input [1] = (“software courses identifier”, “software courses data”)
structural_unit_product: Array [1..2] of Product where
  structural_unit_product [1] = (“listing of  software courses”)
structural_unit_resource: Array [1..2] of Resource where
  structural_unit_resource [1] = (“MySQL management system”, “web interface”)
structural_unit_participant: Array [1..2] of Participant where
  structural_unit_participant [1] = “J-énesis_process_people02”
  structural_unit_participant [2] = “J-énesis_process_people03”)
                              …
J-énesis_service01 class instantiates Customer Relationship Management System
functionalities: Array [1..5] of functionality where  …
  functionality [2] = (“Collects, recall and update software courses”, (“reliability/availability ”),   (“reliability/
recoverability”), (“efficiency”), (”usability”), (“portability”), (“maintainability”))
  functionality [3] = (“Collects, recall  and updates customer retention strategies”, (”reliability/availability”),
(“reliability/ recoverability”), (”usability”), (“efficiency”), (“portability”), (“maintainability”))
                              …
```

Figure 9. Processes to Services levels transformation: J-énesis Customer Relationship Management System services adapt *Customer Service* Process

Note that services functionality [2] and [3] have similar quality properties even if they are used by different people; in *Collect, recall and update software courses* and *Collect, recall and update customer retention strategies* only recall is performed by *Customers*; the others are performed by *Customer Service Analysts*. Portability is transparent, for the use of the Java language and usability is required by both kinds of people since the have to change data and consult (maintainability) directly on-line; recoverability is desired for fault tolerance and efficiency is provided by the network context.

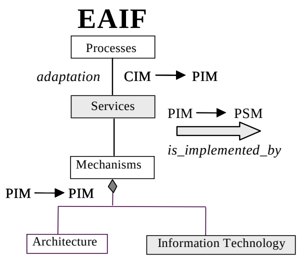


```
J-énesis_service01 class instantiates Customer Relationship Management System
functionalities: Array [1..5] of functionality where
                              …
  functionality [2] = (“Collects, recall and update software courses”,
(“reliability/availability ”), (“reliability/ recoverability”), (“efficiency”),
(”usability”), (“portability”), (“maintainability”))
  functionality [3] = (“Collects, recall  and updates customer retention strategies”,
(”reliability/availability ”), (“reliability/ recoverability”), (”usability”),
(“efficiency”), (“portability”), (“maintainability”))
                              …
J-énesis_mechanism class instantiates Mechanism
integration_method = (“message passing”, “GUI ”)
communication_model = (“synchronous”)
quality = (“J2EE compliant”, “portability”,“ interoperability”,
“maintainability”)
architecture: J_énesis_architecture
IT : J-énesis_information_tecnology

J-énesis_architecture class instantiates Architecture
description = “J-énesis components and connectors”
architecture_style = “layers”
problem_context = “customer relationship management system”
quality = “maintainability”
patterns = {Pattern01, Pattern02}

Pattern01 class  instantiates Persistent Data Manager
pattern_name = Persistent Data Manager
problem_description = (Rogers, 1997)
solution = (Rogers, 1997)
quality = {“reliability ”, “maintainability ”}
```

Figure 10. Services to Architecture levels transformation: the J-énesis Customer Relationship Management System services implementation uses the *Persistent Data Manager* pattern



```
J-énesis_service01 class instantiates Customer Relationship Management System
functionalities: Array [1..5] of functionality where
                              …
  functionality [2] = (“Collects, recall and update software courses”,
(“reliability/availability ”), (“reliability/ recoverability”), (“efficiency”),
(”usability”), (“portability”), (“maintainability”))
  functionality [3] = (“Collects, recall  and updates customer retention strategies”,
(”reliability/availability ”), (“reliability/ recoverability”), (”usability”),
(“efficiency”), (“portability”), (“maintainability”))

J-énesis_mechanism class instantiates Mechanism
integration_method = (“message passing”, “GUI ”)
communication_model = (“synchronous”)
quality = (“J2EE compliant”, “portability”,“ interoperability”, “maintainability”)
architecture: J_énesis_architecture
IT : J-énesis_information_tecnology

J-énesis_information_tecnology class instantiates Information Technologies
hardware = { windows PC’s}
software = {Java, HTML, Java Web Server Tomcat 4, Java Servlets 2.3, Java
Server Pages (JSP) 1.2 , JavaMail 1.2}
communications = {HTTP, TCP/IP, JDBC 2.0 }
database = {MySQL}
```

Figure 11. Services to Information Technology levels transformation: the J-énesis Customer Relationship Management System services implementation uses *Windows PC’s*, *Java*, etc

4. CONCLUSIONS

EAIF models the main elements of an organization aiming to elaborate an integration project or to describe the existing situation of an enterprise with respect to people, processes, services and mechanisms and it is specified as a standard UML model. It is used as a tool to document and identify precisely central aspects of integration platforms. All the EAIF levels can be easily mapped to the MDA models and the corresponding transformations were shown. MDA has enriched the original framework, providing a different structure to the EAIF levels, grouping them into standard-based CIM, PIM and PSM, allowing traceability among the models and increasing the legibility of the framework. Moreover, a brief overview of the known frameworks of Zachman, Stojanovic, and Cummins and their mapping to the MDA models were also shown. The case study is a CRM application which is central to implement an e-business strategy. It is a running system, part of a complex integration research project. We showed precisely all the transformations among the MDA models corresponding to the EAIF levels. In the future, a more formal approach to the EAIF specification is envisaged, as the use of a constraint language or some UML profiles, to increase suitability. We hope that this work is a contribution to practitioners to provide them with a useful tool for modeling engineering.

References

[1] Acuña G. and R. Rodríguez. Desarrollo de un Framework para Soluciones CRM en la Fase de Fidelización del Cliente. License Thesis, Carabobo University, Venezuela, (2003).

[2] Brown A., D. Carnery, E. Morris, D. Smith, and P. Zarrella. Principles of Case Tool Integration, Software Engineering Institute, Oxford University Press (1994).

[3] Buschmann F., R. Meunier, H. Rohnert, S. Peter, and S. Michael. A System of Patterns, John Wiley & Sons Ltd. (1996).

[4] CIO (Chief Information Officer) Council, A Practical Guide to Federal Enterprise Architecture, Version 1.0, February (2001).

[5] Cummins, F. Enterprise Integration, Wiley Computer Publishing (2002).

[6] Flater D. Impact of Model-Driven Standards, Proceedings of the 35th Hawaii International Conference on System Sciences-(HICSS-35.02), 0-7695-1435-9/02, IEEE (2002).

[7] Frankel D. S. Model Driven Architecture: Applying MDA to Enterprise Computing. John Wiley & Sons (2003).

[8] Frankel D. S. Applying EDOC and MDA to the RM-ODP Engineering and Technology Viewpoints, An Architectural Perspective, Version 01-00 http://www.net.intap.or.jp/e/odp/odp-techguide.pdf, (2003).

[9] Frankel D. S., P. Harmon, J. Mukerji, J. Odell, M. Owen, P. Rivitt, M. Rosen, and R. Soley. The Zachman Framework and the OMG's Model Driven Architecture, Business Process Trends WHITEPAPER (2003).

[10] Harmon P. The OMG's Model Driven Architecture and BPM, Business Process Trends, Volume 2, No. 5 (2004).

[11] ISO/IEC 10746-1. International Standard Organization, Information Technology, “Basic Reference Model of Open Distributed Processing” (1998).

[12] ISO/IEC 9126-1. International Standard Organization, “Software Engineering - Product Quality. Part 1: Quality Model”, (2001).

[13] Krzysztof C. and S. Helsen. Classification of Model Transformation Approaches, OOPSLA’03 Workshop on Generative Techniques in the Context of Model-Driven Architecture, (2003).

[14] Laudon K. and J. Laudon. Management Information Systems, Eighth Edition, Prentice Hall (2004).

[15] Losavio F., D. Ortega, and M. Pérez. Modeling EAI, Proceedings of the XXII International Conference of the Chilean Computer Science Society (SCCC 2002), IEEE Computer Society Press, Copiapo, Atacama, Chile, (2002), pp. 195-203.

[16] Losavio F., D. Ortega, and M. Pérez. Towards a Standard EAI Quality Terminology, Proceedings XXIII International Conference of the Chilean Computer Science Society (SCCC 2003), IEEE Computer Society Press, Chillán, BÍO-BÍO, Chile (2003), pp. 119-129.

[17] Losavio F., D. Ortega, and M. Pérez. Comparison of EAI Frameworks to appear in Journal Object of Technology, May/June (2005).

[18] Losavio F., Ortega D., Pérez M., González M., Emad N., Reporte interno de Universidad de Versailles. Modeling Problem Solving Environments Using EAI Concepts. Disponible en http://www.prism.uvsq.fr/rapports/2005/document_2005_78.pdf

[19] Mascolo C., Capra L. and Emmerich W. Mobile Computing Middleware, Advanced Lectures in Networking, LNCS 2497, (2002).

[20] McKeen J. and Smith H. New Developments in Practice II: Enterprise Application Integration, Communications of the Association for Information Systems, Vol. 8, (2002), pp. 451-466.

[21] O'Brien, J. Management Information Systems, Sixth Edition, McGraw-Hill, (2004).

[22] Object Management Group Inc. MDA Specifications. Copyright 1997-2004 http://www.omg.org/mda/specs.htm, (2001).

[23] Object Management Group Inc. Unified Modeling Language, Specification, March, Version 1.5, formal/03-03-01, http://www.omg.org/docs/formal/03-03-01.pdf, (2003).

[24] Open Group, The Open Group Architecture Framework (TOGAF)-Version 7, November (2001).

[25] O'Rourke C., N. Fishman, and W. Selkow. Enterprise Architecture Using the Zachman Framework. Thomson Course Technology (2003).

[26] Sandoe K., G. Corbitt, and R. Boykin. Enterprise Integration, California State University, Chico. John Wiley & Sons, Inc. (2001).

[27] Shaw M. and D. Garlan. Software Architecture - Perspectives on an Emerging Discipline, Prentice Hall (1996).

[28] Stojanovic Z. and A. Dahanayake, and H. Sol. Integration of Component-Based Development Concepts and RM-ODP Viewpoints, 1st. International Workshop on Open Distribute Processing Enterprise, Computation, Knowledge, Engineering and Realisation, WOODPECKER 2001, (2001), pp. 98-109.

[29] Stojanovic Z. and A. Dahanayake. Components and Viewpoints as Integrated Separations of Concerns in System Designing, Workshop on Aspect-Oriented Design (in conjunction with the 1st International Conference on Aspect-Oriented Software Development), Enschede, April 22-26 (2002).

[30] Szyperski C. Component Software. Beyond Object-Oriented Programming, Second Edition, Addison Wesley (2002).

[31] Tiwana A. The Essential Guide to Knowledge Management, Prentice Hall (2001)

[32] Turban E., R. Rainer, and R. Potter. Introduction to Information Technology, third edition, John Wiley & Sons, INC. (2005).

[33] Zachman J. A framework for Information System Architecture, IBM System Journal, Vol. 26, No. 3, (1987), p. 276-292.