Diseño y Consultas de Aplicaciones Hipermedias

Silvia Gordillo

Comisión de Investigaciones Científicas de la Pcia. de Bs. As, y LIFIA,
Universidad Nacional de La Plata.
Argentina
gordillo@info.unlp.edu.ar

Alicia Díaz

Consejo Nacional de Investigaciones Científicas y Técnicas, y LIFIA,
Universidad Nacional de La Plata.
Argentina
alicia@info.unlp.edu.ar

Abstract

En este artículo proponemos un modelo Orientado a Objetos para el diseño de aplicaciones hipermedia que toma las ventajas provistas por el paradigma tales como tipos complejos, definidos por el usuario, atributos no atómicos y no convencionales, etc. De esta forma permitimos no solamente la modelización de la información sino también, formas alternativas para acceder a la misma.

También se presenta un lenguaje de consultas que combina características de los lenguajes de bases de datos orientadas a objetos, con las primitivas de navegación de los hipermedias. El lenguaje definido ofrece tanto la posibilidad de realizar consultas a la información almacenada, como a la información del esquema de la aplicación.

1. Introducción

La tecnología hipermedia a emergido como una excelente alternativa para la construcción de sistemas de información complejos. Esta tecnología ha sido recientemente aplicada en diferentes dominios tales como educación, museos virtuales, bases de datos heterogéneas, ingeniería de software, etc.

Sin embargo, esta tecnología por ser algo reciente, presenta algunos problemas aún no resueltos. Uno de estos problemas se basa en la naturaleza de los hipermedia, y está directamente relacionado con su estructura subyacente de *nodos* y *links,* la cual permite la navegación natural de la información. Si bien esto puede resultar una ventaja en muchos aspectos, también es cierto que conocidos conceptos de la ingeniería de software, como calidad, reusabilidad, modificabilidad no pueden ser aplicados. Por otra parte, la navegación como principal método de acceso, resulta eficiente cuando el usuario tiene un conocimiento general de la información , pero puede causar algunos inconveniente si se desea acceder a información específica, e incluso puede causar desorientación, sobre todo cuando se esta trabajando en un gran hiperespacio.

En este sentido, existen algunas propuestas en las que se definen modelos para el diseño de hipermedias, como son [6], [7], [13], [15], que se describen metodologías de diseño de aplicaciones hipermedias, pero no tratan específicamente el tema de las consultas a estas aplicaciones.

Existen también, algunos enfoques que tratan de agregar las capacidades de consultas a aplicaciones hipermedias como una posibilidad adicional al acceso navegacional. Algunos de estos enfoques usan las bases de datos relacionales como una forma de proveer conocimiento sobre la estructura del grafo y los tipos de nodos. Por ejemplo, [16] combina un sistema de hipermedia con un DBMS, y de esta manera representa la información a través de atributos atómicos que pueden ser consultados. [12] propone una base de datos Orientada a Objetos como soporte para la aplicación hipermedia, con lo cual la información es accedida usando el lenguaje de consulta de la base de datos.

En cambio el objetivo del presente trabajo es desarrollar un modelo que agregue los conceptos mencionados en el párrafo anterior estructurando la información de manera de obtener un esquema de la aplicación que sirva como base, tanto para el diseñador contando con herramientas que le permitan un cierto nivel de comprensión de la estructura de la aplicación; como para el usuario permitiéndole el acceso a la información no solamente a través de la

navegación, sino también por medio de un lenguaje de consultas. La ventaja del modelo es que pone el énfasis en conservar la flexibilidad propia de este campo de manera de preservar las facilidades que provee, como por ejemplo manejar distintas perspectivas de la información, tener una interfaz multimedial, facilidades de navegación, etc.

En la sección 2 describimos las principales características del modelo y los diferentes niveles de diseño para construir aplicaciones hipermedias. En la Sección 3 se describe el lenguaje de consultas definido y en la sección 4 se discuten las principales ventajas del modelo.

2. Un modelo OO para diseñar aplicaciones hipermedias

El modelo propuesto se basa en el paradigma orientado a objetos ya que permite definir un modelo donde datos no-convencionales, como los usados en hipermedias, pueden ser representados en forma natural e implementarse fácilmente las ventajas mencionadas anteriormente [8], [9], [11], [4]. Consiste básicamente de dos niveles de diseño, el primero captura el dominio de la aplicación en un alto nivel de abstracción, donde se definen globalmente las entidades y relaciones; en el segundo nivel de diseño, se definen las perspectivas de los datos, los de navegación y las interfaz. En la sección 2.1 y 2.2 describimos en detalle cada uno de estos niveles.

2.1 *Diseño de alto nivel*

El objetivo de este paso es diseñar la aplicación hipermedia en una forma abstracta dejando de lado los detalles y concentrándose solamente en los elementos más importantes y en sus relaciones. Este mecanismo es similar al proceso de diseño propuesto en [14], el cual distingue las clases de entidades de la aplicación, organizadas en jerarquías de especialización y composición, y las relaciones conceptuales que existen entre ellas. En este nivel solo se describen los objetos de la aplicación y las relaciones, junto con sus propiedades y comportamiento, en cambio no se modela ninguna información acerca de las características de navegación, objetos de interfaz y detalles de implementación.

2.1.1 Entidades

Cada tipo de entidad en el dominio de la aplicación es modelado como una clase de nodo, y se define por un conjunto de propiedades y comportamiento. Estas clases de nodo se relacionan a través de las relaciones de subclase (*es_un*) y de composición (*es_parte_de*), definiendo dos jerarquías de clases independientes.

Un atributo (propiedad) se define por su nombre y su dominio. Los atributos son clasificados, de acuerdo a su dominio, en siete categorías: a*tributo atómico*: su dominio es una clase primitiva (en nuestro modelo texto, sonido, imagen, video, etc. son clases primitivas); a*tributo complejo*: toman valor en clases definidas por el usuario; a*tributo compuesto*: su dominio es una clase no-primitiva y soporta la semántica de la relación *es_parte_de; atributo simple*: es instanciado con un solo valor del dominio; *atributo multivaluado*: es definido a través del los constructores *set_of* o *list_of; atributo multidominio*: puede tomar valor en varios dominios; a*tributo derivado*: toma valor en tiempo de ejecución a través de la evaluación de una función.

Los atributos multidominio se utilizan para implementar diferentes perspectivas de la misma información, ya que permiten visualizar la información a través de diferentes medios.

Existe una clase llamada *Nodo* que es la raíz de la jerarquía de clases de nodos y que soporta la relación *es-un*.

2.1.2 Relaciones

Las relaciones modelan las relaciones conceptuales entre dos o más entidades de la aplicación. Desde el punto de vista de una aplicación hipermedia, modelar las relaciones sirve para permitir al usuario navegar naturalmente a través de la información.

En este modelo las relaciones son objetos y por lo tanto se definen a través de clases que son organizadas en una jerarquía de especialización/generalización, donde la clase raíz se llama *Link*.

La clase *Link* define algunos atributos que sirven para especificar las clases de nodos fuente (atributo *From*) y destino (atributo *To*), cardinalidad y comportamiento para la navegación. Una clase de relación hereda de la clase *Link* y probablemente agregue sus propios atributos.

Es importante destacar que el modelo aquí propuesto distingue entre dos tipos de relaciones navegables, aquellas que describen relaciones particulares entre objetos y la relación *es_parte_de*, la cual determina composición. La composición es un caso especial, ya que define una semántica de navegación particular, permitiendo navegar un

objeto compuesto como una unidad o cada una de sus componentes a partir de su raíz. Este tipo de relación es tratada en otros modelos de hipermedia como [3].

2.2 *Diseño de bajo nivel*

En este nivel el objetivo es:

- diseñar la estructura navegacional definiendo los anchors
- definir perspectivas para visualizar la información, y
- describir los detalles para la interfaz.

Los anchors son los que permiten definir los distintos caminos de navegación que existen a partir de un objeto particular.

Una de las características que, como mencionamos anteriormente debe conservarse cuando se define un modelo para aplicaciones hipermedia, es la de poder definir distintas perspectivas para la misma información. Estas distintas perspectivas de definen por medio del concepto de *Ejemplar* [10]. Cada clase de nodo tiene asociado al menos un ejemplar.

2.2.1 **Ejemplar**

Un ejemplar se define por una lista de atributos que son seleccionados desde aquellos existentes en la clase de nodo asociada; mas el comportamiento esperado. Además, estos tienen los anchors correspondientes a los links que se visualizarán en la perspectiva que implemente este ejemplar.

Cuando se construye un ejemplar se elige sólo un dominio para aquellos atributos multidominio, es decir, se decide cuál va a ser el dominio del atributo para ese ejemplar.

De la misma manera que las clases de nodos y links, los ejemplares se organizan en una jerarquía de especialización/generalización donde la clase raíz se llama *Ejemplar*.

Los ejemplares podrían agregar nuevas relaciones para modelar relaciones excepcionales entre ejemplares. Estas clases de relaciones se modelan vía clases de links, pero donde el dominio de los atributos to y from son clases de ejemplares. Esencialmente, esta facilidad permite navegar las distintas perspectivas de la misma información.

Los dos niveles de diseño previamente descriptos definen el esquema del modelo, es decir el esquema se compone de tres jerarquías: clases de nodos describiendo las entidades de la aplicación, clases de links para las relaciones y clases de ejemplares, las que describen las diferentes perspectivas y los anchors. Este esquema es la base para el lenguaje de consulta que se presentará en la próxima sección.

Para ejemplificar nuestro modelo nos basaremos en la aplicación "*The Way Things Work of DK*" (*). esta es una aplicación educativa donde el usuario puede navegar para obtener aspectos tales como: cómo funciona una máquina, sus principios básicos asociados, la historia del invento hasta la biografía de su autor. La figura 1 muestra el esquema de esta aplicación, donde se especifican algunas de las entidades y relaciones encontradas en ella. Las relaciones descriptas a través de un diamante identifican la relación *es_parte_de*.

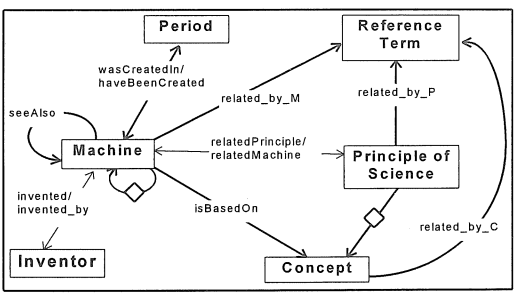

Figura 1: diseño de E-R de la aplicación The Way Things Work.

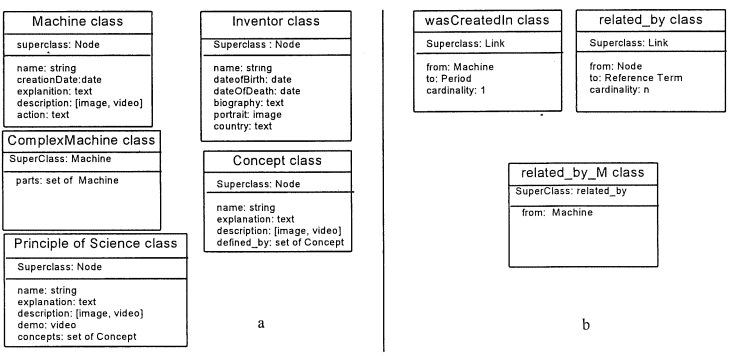

Figura 2: a.Ejemplos de clases Nodos b.Ejemplos de clases Link

En la figura 2 mostramos las clases de nodos y links para el ejemplo anterior, donde el lector puede ver sus atributos. En particular la Figura 2.a permite observar las clases de nodos, donde la clase de nodo *ComplexMachine* es subclase de la clase *Machine*.

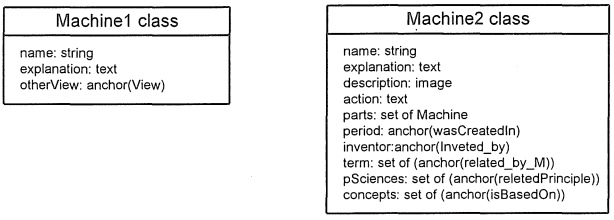

Figura 3: Dos ejemplares para la clase de nodo Machine.

La figura 3 muestra dos perspectivas posibles para el nodo Machine, la primera con el nombre e información general, mas un anchor al dibujo de la máquina y la segunda con información detallada mas la imagen de la misma.

3. Consultas a hipertextos

Con las bases de las definiciones previas, y a partir de la construcción del esquema , es posible definir un lenguaje de consultas del estilo de los lenguajes de consultas de las bases de datos pero además con la capacidad de navegación mas características propias de las aplicaciones hipermedias.

Los usuarios pueden acceder a la información vía dos formas: modo navegación o modo consulta. La primera es la manera tradicional de acceder a las aplicaciones hipermedias, en el segundo modo, en cambio, es posible acceder tanto a la información contenida en la aplicación como también a la información del esquema consultando alguna de las tres jerarquías de clases (*Nodos*, *Link* y *Ejemplar*).

El modelo de consultas está basado en el modelo definido por [8], donde se han realizado algunos cambios necesarios para considerar las características propias de las aplicaciones hipermedias.

3.1 *Consultando la Información*

Este tipo de consultas son orientadas al usuario, ya que estas permiten recuperar la información almacenada, donde el operador más importante es la *selección de objetos*. También existen otros constructores para manejar objetos compuestos, relaciones y propiedades (a través del operador *project*)

3.1.1 Operación de Selección

La selección de objetos desde una o más clases, cuando una condición es verdadera, es similar a la selección tradicional de las bases de datos relacionales.

***select** [ListaClases] **from** AsocVarObj **in** [fuente] [**where** condición]*

donde:

ListaClases: es la especificación de las clases resultantes.

AsocVarOobj: indica la asociación de las variables a los conjuntos de instancias.

fuente: es el hipertexto o el conjunto de objetos a ser consultados.

condición: es una combinación de predicados booleanos..

[...]: indica parámetros opcionales.

El resultado de esta consulta es un hipertexto formado por las clases que aparecen en la lista de clases resultantes más la relaciones que existen entre ellas (en otras palabras el resultado es un subhipertexto creado, a partir del hipertexto original, con solamente aquellas clases que aparecen involucradas en la consulta junto con las relaciones entre ellas).

El siguiente ejemplo recupera los inventores nacidos antes de 1900 y las máquinas que inventaron.

*Q := **select** Machine, Inventor **from** Inventor :i, Machine :m **where** (i.dateofBirth < 1900)*

3.1.1.1 *Cuantificadores*

La condición puede incluir atributos multivaluados, por esta razón el modelo incluye dos cuantificadores: EACH y EXIST, que corresponden a los cuantificadores universal y existencial respectivamente.

El siguiente ejemplo recupera maquinas que usan un motor eléctrico.

*Q := **Select** Machine **from** Machine :m **where exits** m.parts = "electrical engine"*

3.1.1.2 *Operadores sobre Conjuntos*

El predicado de selección puede involucrar atributos multivaluados. Para describirlos usamos los operadores SET y LIST [8], los que nos permiten trabajar con conjuntos y listas de atributos.

3.1.2 Operación de Proyección

La operación de proyección permite seleccionar algunos atributos de objetos pertenecientes a una clase. El formato de la operación es:

select *[ListaClases]* ***from*** *AsocVarObj* ***in*** *[fuente]* *[**where** condición]*

donde la definición de *ListaClases, AsocVarObj, fuente y condición* es la misma que en la operación de selección.

El resultado de esta consulta es siempre un conjunto de listas con los atributos seleccionados.

En el siguiente ejemplo, el resultado es una lista con los nombres de inventores nacidos antes de 1900.

Q := select *Inventor.name* ***from*** *Inventor :i* ***where*** *(i.dateofBirth < 1900)*

3.1.3 Operación Related_By

Esta operación permite establecer si existe una relación entre dos objetos nodos. El resultado de la operación es un valor booleano que es verdadero cuando la relación existe y falso en caso contrario.

La sintaxis de esta consulta es:

related_by *(ObjetoFuente, NombreRelación, ObjetoDestino)*

El ejemplo permite recuperar los inventores de aquellas máquinas creadas durante la revolución industrial.

Q := select *Inventor, Machine* ***from*** *Machine:m, Period :p* ***where*** *(p.name= "Industrial Revolution" and* ***related_by*** *(m, WasCreatedIn, p))*

3.1.3.1 *Operación de Asignación*

A través de esta operación se permite almacenar el resultado de las consultas. Esta operación es la base que nos permite realizar nuevas consultas sobre el resultado obtenido de una previa.

3.2 *Consultando el esquema*

Este tipo de consultas, si bien pueden ser útiles para el usuario, están especialmente formuladas para asistir al diseñador de una aplicación, de manera por ejemplo, que pueda obtener información sobre la jerarquías del esquema del hipertexto.

3.2.1 Operaciónes sobre jerarquías

La operación ***Hierarchy*** es usada para establecer cuales son las superclases o subclases de una clase dada.

hierarchy↑ *NombreClase* ***From*** *Jerarquía*

or

hierarchy↓ *NombreClase* ***From*** *Jerarquía*

donde:

NombreClase: es el nombre de una clase de alguna de las jerarquías del esquema.

Jerarquía: es el nombre de algunas de las jerarquías *Nodo*, *Link* o *Ejemplar*

La diferencia entre **hierarchy**↑ y **hierarchy**↓ es que la primera recupera nombres de superclases, en cambio la segunda nombres de subclases.

3.2.2 Operaciones sobre clases

La operación ***properties*** recupera las propiedades definidas en una clase.

properties *NombreCklase* ***from*** *Jerarquía*

NombreClase y Jerarquía se definen de la misma manera que para la operación **Hierarchy**.

Las operaciones ***source*** y ***traget*** recuperan, respectivamente, las clases de nodos origen y el destino de una relación particular.

source *NombreClaseLink*

target *NombreClaseLink*

El operador **exemplar** permite conocer los nombres de los ejemplares asociados a una clase de nodo particular.

exemplar *NombreClaseNodo*

Existen dos operaciones ***related_to*** y ***related_from*** que permiten determinar respectivamente, qué clases de nodos son alcanzables desde una clase específica y todas las clases de nodos desde las cuales es alcanzable una clases particular.

related_to *NombreClaseNodo*

related_from *NombreClaseNodo*

4. Conclusiones

Hemos presentado un modelo para aplicaciones hipermedias, el cual provee las bases para estructurar información a partir de los conceptos de Nodos, Relaciones y Ejemplares. Usando este modelo se puede diseñar una aplicación en diferentes niveles de abstracción, de la misma forma que ocurre en los métodos de ingeniería de software modernos, sin perder la facilidades tradicionales de los hipermedias. El mantenimiento de las aplicaciones diseñadas con este modelo resulta mucho mas simple que cuando se usa el enfoque tradicional porque se permite un mejor entendimiento del dominio de la aplicación.

La posibilidad de definir un esquema y un lenguaje de consultas incrementa las características para manipular aplicaciones hipermediales y provee herramientas adicionales para recuperar información, eliminando de alguna manera la desorientación de los usuarios. Esta capacidad también es muy útil para los diseñadores ya que pueden consultar el esquema aumentando su conocimiento sobre el dominio de la aplicación.

(*) The Way Things Work es trademark de Dorling Kindersley MULTIMEDIA.

5. Referencias

[1] B. Amann, M. Scholl. *"Gram: A graph data model and query language"*.Proceedings of the 4th ACM Conference on Hypertext and Hypermedia (ECHT'92), Milano, Italy, December 1992

[2] B. Amann, M. Scholl. *"Quering typed hypertexts in Multicard/O2"*. Proceedings of the 5th ACM Conference on Hypertext and Hypermedia (ECHT'94), Edinburgh, September, 1994.

[3] V. Chistiphides, A. Rizk. *"Quering structured documents with hypertext links using OODBMS"*. Proceedings of the 5th ACM Conference on Hypertext and Hypermedia (ECHT'94), Edinburgh, September, 1994.

[4] D. Chorafas and H. Steinmann. "*Object Oriented Databases*". PTR Prentice Hall, 1993

[5] M.P. Concens, A. Mendelzon. *"GaphLog: a visual formalism for real life recursion*". Proceedings of the ACM SIGACT-SIGMOD Symp. on Principles of Database Systems, pp 404-416, Nashville, Tennessee, 1990.

[6] F. Garzotto, P. Paolini, D. Shwabe, *"HDM - A Model Based Approach to Hypermedia Applications Design"*, ACM Trans. Info. Syst. 11, 1 (Jan.1993) pp 1-26.

[7] T. Isakowitz, E. A. Stohr, P. Balasubramaninan, *"RMM: A Methodology for Structured Hypermedia Design"*, Communications of the ACM, August 1995.

[8] W. Kim. *"Introduction to Object Oriented Databases"*, MIT Press, 1990.

[9] W. Kim, *"Modern Database Systems"*,. ACM Press, 1995.

[10] W. LaLonde, *"Designing Families of Data Types Using Exemplars"*, ACM Toplas, April 1989.

[11] M. Loomis. *"Object Oriented Databases. The Essentials"*. Addison Wesley Publishing Company, 1994

[12] M. Marmann and G. Schlargeter. *"Towards a Better Support for Hypermedia Authoring: The HYDESIGN Model"*. Proceedings of the 4th ACM Conference on Hypertext and Hypermedia (ECHT'92), Milano, Italy, December 1992

[13] J. Nanard, M. Nanard. "*Using structured types to incorporate knowledge in hypertext*". Proceedings of Hypertext'91, pages 329-343, December 1991.

[14] J.J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, W. Lorensen. *"Object Oriented Modeling and Design"*. Prentice Hall Inc., 1991.

[15] D. Schwabe, G. Rossi, S. Barbosa. *"Systematic Hypermedia Design with OOHDM"*. Proceedings of the Twenty-Eighth Hawaii International Conference on System Sciences, Maui, Hawaii (Jan. 1995).

[16] H. Van Dyke Parunak. *:Don't link me in: Set based hypermedia for taxonomic reasoning"*. Proceedings of Hypertext'91, pages 233-242, December 1991.