Aplicando Restricciones en un Datawarehouse Temporal Utilizando UML/OCL

Carlos Neil[1] y Claudia Pons[2]

[1] Universidad Abierta Interamericana
Facultad de Tecnología Informática
Carlos.Neil@vaneduc.edu.ar

[2] LIFIA - Universidad Nacional de La Plata
Buenos Aires, Argentina
cpons@info.unlp.edu.ar

Resumen

Un datawarehouse es una colección de datos no volátiles, variables en el tiempo y orientado a un tema específico utilizado para tomar decisiones. La necesidad de registrar valores que permitan evaluar tendencias, variaciones, máximos y mínimos, justifican considerar en el diseño la evolución temporal de atributos o interrelaciones. La determinación de restricciones que permitan mantener la consistencia de los datos almacenados e impidan los solapamientos de rangos temporales, permiten conservar la integridad de la base de datos. En la etapa de modelado la determinación de estas restricciones se realizan de manera informal. Establecer sin ambigüedad las mismas no es posible sin un lenguaje apropiado. Proponemos determinar restricciones en el modelo de datos transformando el esquema multidimensional temporal a UML y emplear OCL para documentar dichas limitaciones.

Palabras claves: Datawarehouse Temporal, estereotipos, UML, OCL.

1. Introducción

Un datawarehouse es una colección de datos no volátiles, variables en el tiempo y orientado a un tema específico que se utiliza para tomar decisiones organizacionales. Los datos históricos son más importantes que los detalles y los registros individuales. En el datawarehouse, las dimensiones determinan la granularidad para la representación de hechos, las jerarquías en las dimensiones indican cómo las instancias de hechos pueden ser agrupadas y seleccionadas para los procesos de toma de decisión (Agrawal et al. 1997). Se supone implícitamente que las dimensiones son independientes entre sí; sin embargo esto no es cierto, una o más de éstas dependen de la dimensión tiempo (Bliujute et al. 1998). En el datawarehouse el tiempo es una de las dimensiones para el análisis (Chaudhuri, Dayal. 1997; Golfarelli et al. 1998), pero éste hace referencia al momento en que se realizó una transacción, no se detalla cómo varían los atributos o interrelaciones involucradas en ellas. La necesidad de registrar valores que permitan evaluar tendencias, variaciones, máximos y mínimos, justifican considerar en el diseño del datawarehouse cómo ciertos atributos o interrelaciones pueden variar en el tiempo.

Cuando consideramos aspectos temporales en las bases de datos transaccionales debemos registrar dos tipos de información para los atributos, de modo tal de poder capturar completamente la evolución de éstos: el tiempo válido de un hecho, que es aquel en el cual el hecho es verdadero en el universo de discurso y el tiempo de transacción, que representa el momento en el cual la información se registra en el sistema (Mc Brien et al. 1992; Jensen et al. 1994; Gregersen, Jensen.

1998). Se puede representar la estructura temporal utilizando una extensión del modelo entidad-interrelación que permita la representación del tiempo válido (bases de datos históricas) o del tiempo válido y el tiempo de transacción (bases de datos bitemporales) (Mc Brien et al. 1992; Gregersen et al. 1998; Gregersen, Jensen. 1998; Pedersen, Jensen. 1998). Para las actividades de toma de decisión se considera principalmente el tiempo válido (Mc Brien et al. 1992).

Aunque las bases de datos orientadas a objetos comerciales están disponibles, la tecnología de base de datos relacionales usualmente es la preferida para almacenar datos debido a su madurez. Un datawarehouse implementado sobre un sistema administrador de bases de datos estándar relacional se denomina ROLAP (OLAP Relacional). Estos servidores almacenan los datos en una base de datos relacional además de soportar extensiones del SQL, accesos especiales y métodos de implementación para hacer más eficiente el modelo multidimensional y sus operaciones (Chaudhuri, Dayal. 1997). En una arquitectura relacional los datos están organizados en esquemas estrella o copo de nieve; el primero consiste en un tabla de hechos central y varias tablas dimensión denormalizadas, las medidas de interés están almacenadas en la tabla de hecho. La versión normalizada del esquema estrella es el esquema copo de nieve en donde cada nivel de agregación tiene su propia tabla dimensión (Vassilliadis, Sellis. 1999)

Los formalismos gráfico-textuales son representaciones gráficas que además poseen una notación formal para especificar restricciones que se integren como texto dentro de la primera; permiten disponer de precisión sin perder el impacto visual. UML (UML. 2001) se ha establecido en el estándar para el modelado de objetos en las etapas de análisis y diseño de sistemas de software, es un lenguaje visual que se utiliza para especificar, visualizar, construir y documentar artefactos de software. Existen un gran número de herramientas CASE que tienen la facilidad de dibujar y documentar diagramas UML, muchas de ellas ofrecen también generadores automáticos de código e ingeniería inversa (Richters, Gogolla. 2000).

En el modelo orientado a objetos, un gráfico como el de clases no es suficiente para lograr una especificación precisa y no ambigua (Pons et al. 1999, 2000). Existe la necesidad de describir restricciones adicionales sobre los objetos del modelo. Muchas veces esas restricciones se describen en lenguaje natural. La práctica ha revelado que, muy frecuentemente, esto produce ambigüedades; para evitarlos se han desarrollado los lenguajes formales, la desventaja es que, si bien son adecuados para personas con una fuerte formación matemática, son difíciles para el modelador de sistemas promedio. El OCL (OCL. 2001) ha sido creado para cubrir esa brecha, es un lenguaje formal, fácil de leer y escribir y permite expresar información extra sobre los modelos usados en el desarrollo orientado a objetos; es un lenguaje declarativo, no tiene efectos laterales, el estado de un objeto no cambia debido a la evaluación de una expresión OCL. Toda expresión está escrita en el contexto de una instancia de clase definida en un modelo UML y define un conjunto de tipos de operaciones asociadas. Dado un objeto, es posible acceder a sus atributos o invocar sus operaciones mediante el operador punto (.) o, en el caso en que el objeto sea una colección, con el operador flecha (->). Con OCL pueden especificarse invariantes de clase, éstas son expresiones lógicas que han de cumplirse durante toda la vida de las instancias de dicha clase.

En un datawarehouse temporal existen un conjunto de restricciones vinculadas con el dominio de la aplicación que, con frecuencia, no quedan documentadas además de las restricciones propias del modelo relacional subyacente. Cuando se trabaja con rangos temporales, la determinación de restricciones que impidan los solapamientos de los datos en los rangos establecidos por los tiempos válidos constituyen un mecanismo que permite mantener la integridad de los datos almacenados. En la etapa de modelado la determinación de estas restricciones se realizan de manera informal; el establecimiento sin ambigüedad de las mismas no es posible sin un

lenguaje apropiado. Utilizaremos OCL para especificar invariantes sobre clases y, además, como lenguaje de navegación a través del modelo.

Para la utilización del lenguaje de restricciones transformaremos el modelo multidimensional extendido con aspectos temporales (Neil, Ale. 2002), a un diagrama de clases UML y utilizaremos, para adaptar UML a un datawarehouse temporal, el mecanismo de extensión estándar (estereotipos) y OCL.

El resto del trabajo está organizado de la siguiente manera, en el capítulo 2 se presenta un ejemplo motivante; en el capítulo 3 se muestra la transformación de un esquema multidimensional a UML; en el capítulo 4 se definen los atributos y las interrelaciones temporales; en el capítulo 5 se detalla la transformación del modelo multidimensional temporal a UML; en el capítulo 6 se presentan las restricciones en un datawarehouse temporal utilizando OCL; en el capítulo 7 se detallan los trabajos relacionados y, por último, se muestran la conclusión y trabajos futuros.

2. Motivación

Analizaremos los problemas que surgen al considerar un esquema multidimensional en donde las dimensiones representan solamente el tiempo actual. El ejemplo será descrito, en una primera fase, mediante un modelo entidad-interrelación (gráfico 1); luego se presentará un esquema copo de nieve (Golfarelli et al. 1998) del datawarehouse derivado de aquél (gráfico 2).

En la participación de nadadores en competencias deportivas se debe registrar, para cada evento, el tiempo obtenido por el atleta, la fecha de realización del torneo, los datos del competidor y del patrocinante, el club al que pertenece, el puesto obtenido, etc. Por simplicidad omitimos en los diagramas otros atributos.

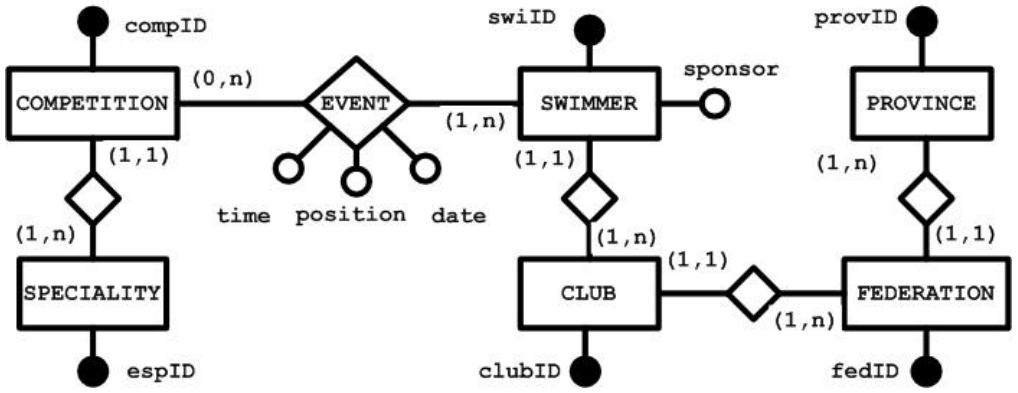

Gráfico 1. Diagrama entidad-interrelación

El hecho principal a analizar en el modelo multidimensional es la prueba o "evento". Los atributos de hecho son el tiempo que realizó el nadador en la prueba y la posición obtenida. Las dimensiones a considerar son la fecha en que se realizó la prueba y el nombre del competidor. Además, un deportista puede pertenecer, en un momento dado, solamente a un club, pero aquél puede cambiar de institución. De la misma forma, un deportista también puede cambiar de patrocinante. Mediante la aplicación del algoritmo para la construcción de un datawarehouse a partir de un diagrama entidad-interrelación (Golfarelli et al. 1998), obtendremos un esquema copo de nieve como muestra el gráfico 2. Este modelo permite realizar sobre él las funciones típicas de un datawarehouse; los problemas de diseño de esta estructura surgirán si quisiéramos, por ejemplo,

determinar cuándo un nadador compitió para un club determinado o cuáles fueron los distintos patrocinantes que un deportista tuvo durante su vida. Observamos que, aunque los datos históricos están almacenados, la forma de recuperarlos requeriría de mecanismos de búsquedas más complejos. Una alternativa que permitiría contemplar las modificaciones temporales sería transformar, como dimensión, a aquellos componentes de la jerarquía que varíen en el tiempo. Sin embargo esta opción solamente reflejaría el cambio, no cuando éste se constituyó como válido.

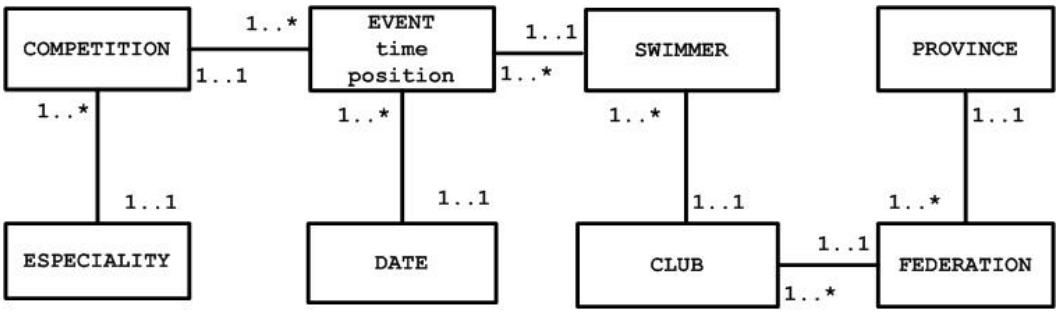

Gráfico 2. Esquema copo de nieve

3. Transformación de un Esquema Multidimensional a UML

UML provee dos mecanismos que potencian su poder expresivo: estereotipos y expresiones OCL. Un estereotipo es un mecanismo de extensión de UML que se usa para clasificar elementos del modelo de manera que se comporten como instancias de nuevas meta clases virtuales basadas en meta clases existentes. Un estereotipo puede introducir valores adicionales, restricciones y una nueva representación gráfica. Por otra parte, OCL es un lenguaje formal declarativo basado en lógica de primer orden. Utilizaremos estereotipos y OCL para establecer distintos tipos de restricciones en una datawarehouse temporal.

La mayoría de los datawarehouse utilizan el esquema estrella para representar el modelo de datos multidimensional. Éste consiste en una tabla para el hecho principal de análisis más una tabla para cada dimensión asociada. Cada tupla en una tabla de hechos tiene un puntero (clave externa) a cada una de las dimensiones. Las tablas dimensión tienen columnas que corresponden a sus atributos. El esquema estrella no provee explícitamente soporte para la jerarquía. El esquema copo de nieve es un refinamiento de aquel, donde la jerarquía se representa explícitamente mediante la normalización de las tablas dimensión. La tabla de hecho tiene, además, un conjunto de valores que son el objeto de análisis. Cada combinación de valores de la dimensión define una instancia de hecho (Chaudhuri. 1997. Vassilliadis. 1999).

Para el modelado con UML utilizaremos los siguientes estereotipos: <<table>> para designar una tabla en el modelo relacional, <<pk>>, para la clave primaria, <<fk>> para la clave foránea y <pkf>> para la clave primaria cuando, además, sea clave foránea (Gornik. 2002a).

La transformación del esquema multidimensional a UML es directa, la tabla de hechos y las tablas dimensión se transformarán en clases con estereotipo <<table>>. La tabla de hecho tendrá, además, como atributos los valores objeto del análisis en el datawarehouse y como identificador único la unión de las claves primarias de cada tabla dimensión asociada, cada una de ellas será clave foránea a su clase dimensión (tabla) respectiva. Las tablas dimensión normalmente tienen atributos que la describen, serán éstos atributos de la clase. Las dimensiones se asocian con sus jerarquías para especificar niveles de agrupamiento específicos y por lo tanto mayor granularidad en la visión de los datos. A partir de las dimensiones, su identificador determina funcionalmente a cada

identificador de la jerarquía, de modo tal que se establecen entre ellos relaciones x-a-uno. Utilizaremos para especificar la jerarquía en las dimensiones la asociación de composición para reforzar el hecho de que la multiplicidad debe ser uno. Cada tabla de la jerarquía se transformará en una clase con el estereotipo <<table>>, sus atributos serán los campos de la tabla, su clave primaria, la de la tabla respectiva y tendrá una clave foránea por cada tabla relacionada x-a-uno con ella. Se pueden utilizar nombres en las asociaciones y roles para clarificar el tipo de relación (se omite en el gráfico). En el gráfico 3 se muestra la transformación del esquema de el gráfico 2.

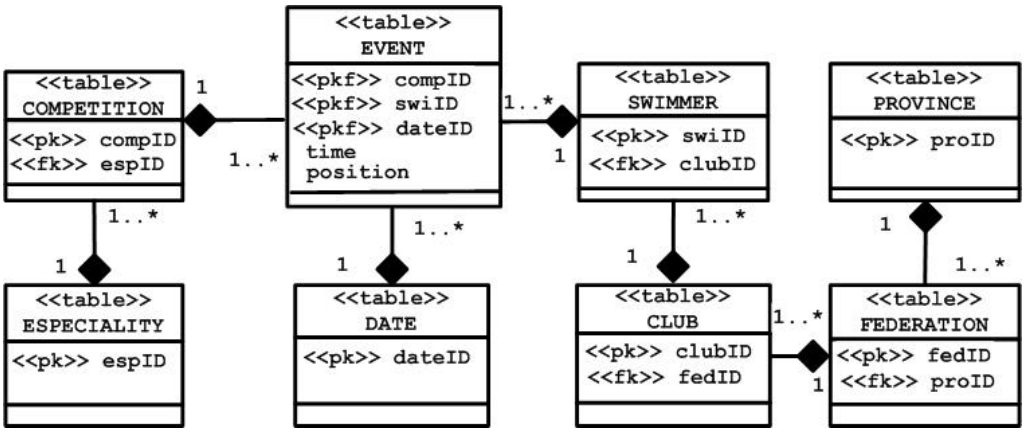

Gráfico 3. Trasformación del esquema copo de nieve a UML

4. Modelo Temporal

Utilizaremos el modelo conceptual temporal que captura la variación de atributos e interrelaciones en función del tiempo (Neil, Ale. 2002). El modelo contempla solamente el tiempo válido. Transformaremos el modelo entidad-interrelación a un diagrama de clases UML.

4.1 Modelo Conceptual Temporal

Def #1: dado un atributo compuesto multivaluado que denominamos *Atributo-T* = {Valor, tiempoInicial, tiempoFinal}, lo proponemos como atributo temporal de una entidad E o una interrelación R. El tiempoInicial y el tiempoFinal determinan el rango dentro del cual el atributo *Valor* es verdadero. En el modelo conceptual presentado, todo atributo que varíe en el tiempo y que, además, resulte necesario conservar sus valores será transformado en *Atributo-T* en su representación en el modelo conceptual. En el gráfico 4 se muestra un ejemplo de atributo temporal.

Def #2: dada una entidad E con atributos $\{A_1,..., A_n\}$, decimos que es una entidad temporal si tiene atributos A_i $(1 \leq i \leq n)$ que son temporales. La denominaremos *Entidad-T*.

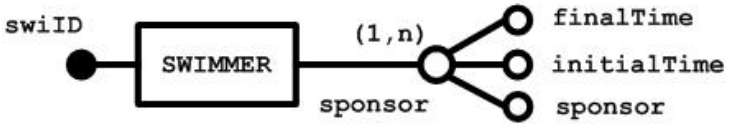

Gráfico 4. Atributo compuesto multivaluado

Una entidad E con *m* atributos temporales puede representarse, en el modelo conceptual, como una entidad regular E con *m* entidades débiles vinculadas, la interrelación que las representa llevará una T. Por ejemplo, si un nadador puede tener más de un sponsor en su vida deportiva, el atributo variante (sponsor) se transformará en *Atributo-T* (gráfico 5).

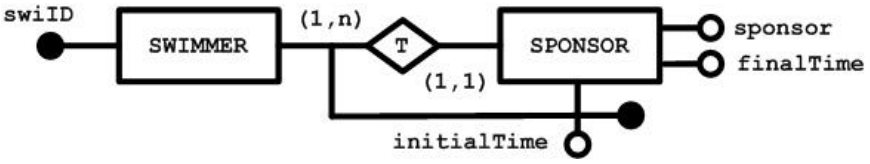

Gráfico 5. Transformación de atributo temporal

Para el modelado con UML utilizaremos el estereotipo: <<tempTable>>, éste indica que la clase (tabla) registra, además, aspectos temporales. La transformación es la siguiente (gráfico 6): a) las entidades regulares se transformarán en clases, con el mismo nombre de la entidad, los atributos indentificadores tendrán el estereotipo <<pk>>, los demás atributos se transformarán directamente en atributos de la clase. b) La entidad débil, que representa el atributo temporal, se transformará en una clase con el estereotipo <<tempTable>> con igual nombre, tendrá como atributo identificador la unión del identificador de la entidad regular más el atributo tiempoInicial, el identificador de la entidad regular tendrá el estereotipo <<pkf>>, el atributo tiempoInicial <<pk>> y, además, los atributos tiempoFinal y valor. La interrelación temporal se transformará en una asociación respetando las siguientes multiplicidades, del lado de la clase (entidad regular) será 1..1, del lado de la clase transformada de la entidad débil, 1..*.

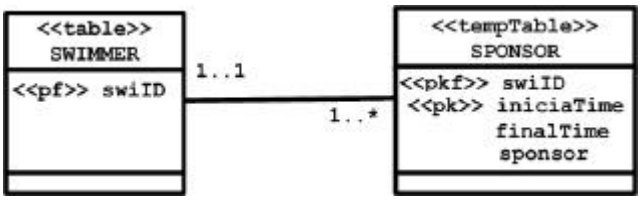

Gráfico 6. Trasformación del atributo temporal a UML

Def #3: dada una interrelación binaria denominada *Interrelación-T,* que vincula a dos entidades E_1 y E_2, con atributos descriptivos tiempoInicial y tiempoFinal, decimos que es una interrelación binaria temporal si el vínculo entre esas entidades puede variar en el tiempo (gráfico 7).

En el modelo propuesto, si una interrelación varía en el tiempo, ésta se denominará *Interrelación-T*, tendrá card-max(E_1, T) = card-max(E_2, T) = n, (n>1) y será graficada con una T.

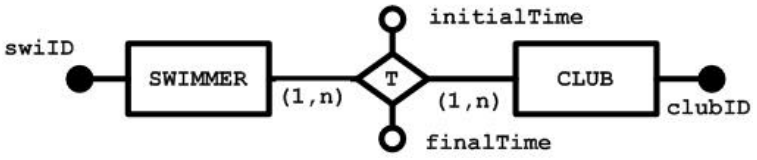

Gráfico 7. Interrelación temporal

La transformación a UML es la siguiente (gráfico 8): a) las entidades regulares se transformarán en clases tal como se determinó en ref #2; se establecerán entre ellas una

asociación con multiplicidad 1..* de ambos lados, b) la interrelación temporal se transformará en una clase asociación con el estereotipo <<tempTable>>, sus atributos serán el identificador de una de las entidades (cualquiera de las dos) más el tiempoInicial, el identificador de la entidad se etiquetará con el estereotipo <<pkf>>, el atributo tiempoInicial con <<pk>>, además tendrá el atributo tiempoFinal.

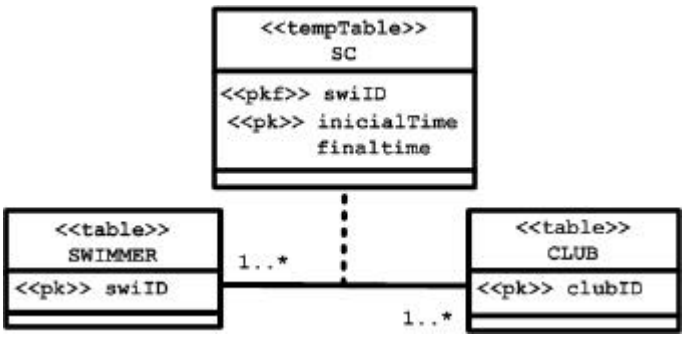

Gráfico 8. Trasformación de la interrelación temporal a UML

Cada atributo, en el modelo relacional, debe tener un dominio subyacente; en el diagrama de clases cada atributo tiene un tipo de datos asociado que representa el dominio de cada campo de la tabla.

5. Transformación del Modelo Multidimensional Temporal a UML

Para la obtención de un esquema multidimensional temporal del modelo de datos presentado en el gráfico 1, utilizaremos el algoritmo presentado en (Golfarelli et al. 1998) y adaptado en (Neil, Ale. 2002). En el gráfico 9 se muestra el esquema multidimensional temporal obtenido.

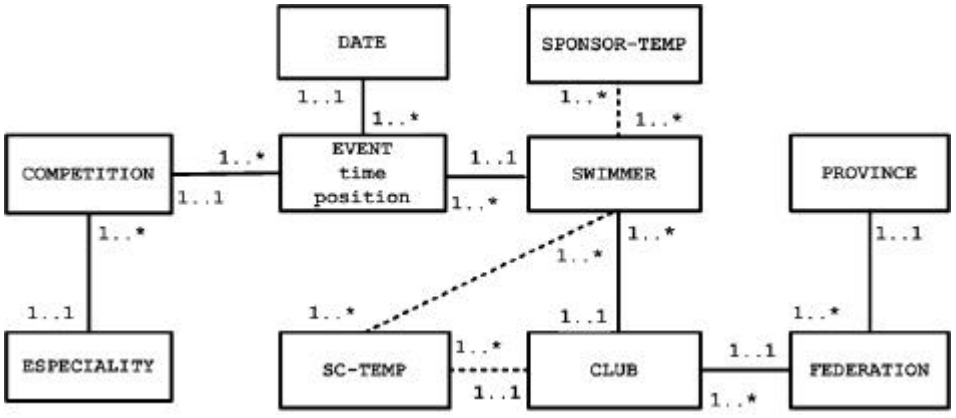

Gráfico 9. Esquema multidimensional temporal

La inclusión de atributos e interrelaciones temporales en el esquema (se vinculan mediante líneas punteadas) precisa de una consideración especial en su transformación al esquema multidimensional: éstos no formarán parte de la jerarquía para las operaciones de roll-up y drill down, solamente permitirán evaluar cómo ciertos atributos e interrelaciones han variado en el tiempo. Constituyen lo que se denomina jerarquías no estrictas (Pedersen, Jensen. 1998).

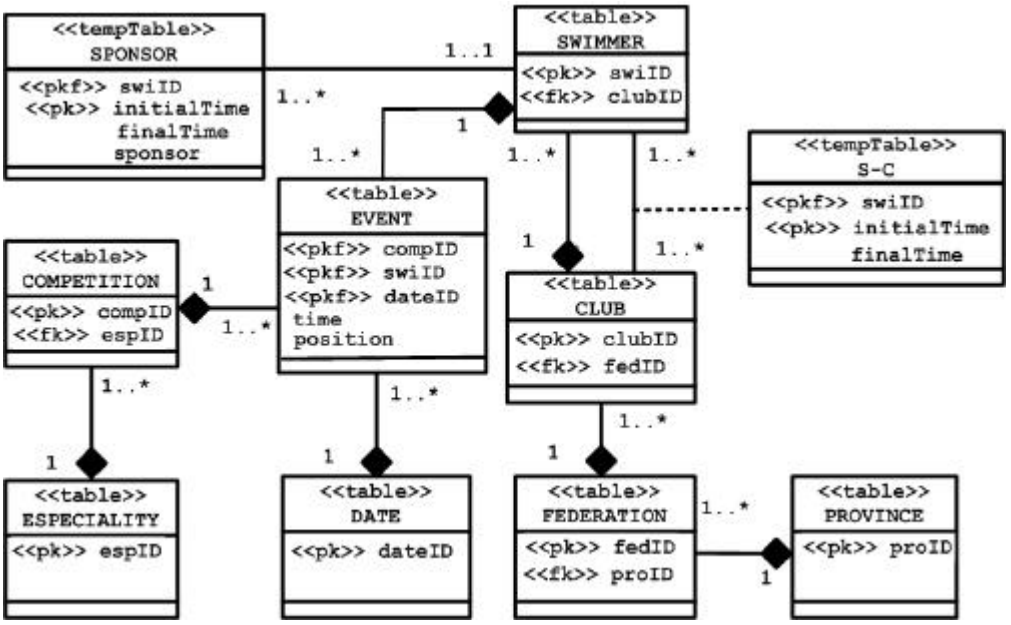

Gráfico 10. Trasformación del esquema multidimensional temporal a UML

Para la transformación del modelo multidimensional a UML se deberán contemplar las jerarquías no estrictas formadas por atributos e interrelaciones temporales. Los primeros se transformarán tal como se expresa en def #2 b); las interrelaciones temporales se transformarán como lo expresa la def #3 b). En este último caso preservamos ambas jerarquías, la instantánea y la temporal. En el gráfico 10 se muestra el modelo multidimensional temporal.

6. Restricciones en un Data Warehouse Temporal Utilizando OCL

El objetivo principal de OCL es especificar restricciones sobre los posibles estados de un sistema con respecto a un modelo dado. Una expresión OCL es declarativa en el sentido que dice *qué* restricción hay que mantener, pero no *cómo* se realiza. Utilizaremos OCL para establecer distintos tipos de restricciones en un data warehouse temporal. Dividiremos a éstas en tres tipos: las restricciones propias de modelo lógico, en este caso relacional (por ejemplo, clave primaria y integridad referencial); las restricciones del dominio de aplicación del datawarehouse y, por último, las restricciones temporales. Daremos algunos ejemplos de esas restricciones, utilizaremos el modelo presentado en el gráfico 10.

6.1 Restricciones del Modelo Relacional

Especificaremos las restricciones propias del modelo relacional mediante la definición de un grupo de estereotipos aplicables a diferentes elementos de UML, de la siguiente forma:

Definición del estereotipo <<pk>> : El estereotipo <<pk>> se aplica a instancias de la metaclase Attribute. Permite identificar a aquellos atributos que actúan como clave primaria (Primary Key) en una tabla relacional.
Definición del estereotipo <<fk>>: El estereotipo <<fk>> se aplica a instancias de la metaclase Attribute. Permite identificar a aquellos atributos que actúan como clave foránea (Foreign Key) en una tabla relacional.

Definición del estereotipo <<pfk>>: El estereotipo <<pfk>> se aplica a instancias de la metaclase Attribute. Permite identificar a aquellos atributos que actúan tanto clave primaria y, además, como clave foránea en una tabla relacional.
Definición del estereotipo <<Table>>: El estereotipo <<Table>> se aplica a instancias de la metaclase Class, permitiendo especificar que una clase representa una tabla en una base de datos relacional.

Para facilitar la definición de las restricciones impuestas por el estereotipo previamente definiremos las siguientes operaciones adicionales:

`1- primaryKeys: Table -> Set (Attribute)`
La operación retorna el conjunto con todos los atributos de la tabla que forman parte de la clave primaria (notar que la clave primaria puede estar formada por un solo atributo o por la concatenación de dos o más), está definida de la siguiente forma:

```
t.primaryKeys = t.attributes -> select(a ¦ a.stereotype = <<pk>>)
```

Por ejemplo: en la tabla de Swimmer existe una clave primaria simple dada por la identificación del nadador, es decir `Swimmer.primaryKeys ={swiID}`, mientras que en la tabla Event la clave primaria esta formada por tres atributos, es decir `Event.primaryKeys = {compID, swiID, dateID}`.

2- `foreignKeys: Table->Set (Attribute)`
La operación retorna el conjunto con todos los atributos de la tabla que actúan como clave foránea y esta definida de la siguiente forma:

```
t.foreignKeys = t.attributes -> select(a ¦a.stereotype = <<fk>>)
```

Por ejemplo: `Swimmer.foreignKeys = {clubID}`

3- `value: Instance,Attribute -> Instance`
La operación retorna el valor correspondiente a un atributo dado en una instancia dada

```
value(i,a)= (i.slot -> selectedAttributeLinks(a)).value
```

Por ejemplo `value(juan,swiID)` podría ser 17470 siendo juan una instancia de Swimmer y 17470 el valor de su identificación.

A continuación incluimos algunos de las restricciones definidas para el estereotipo <<Table>>:

a) Toda tabla debe contener al menos una clave primaria:
```
Context t:Table inv:
not (t.primaryKeys -> isEmpty)
```

b) Dos objetos (tuplas) de una misma clase (tabla) no pueden tener los mismos valores para la clave primaria:
```
Context t:Table inv:
t.instances -> forAll (x,y ¦value(x,k1) = value(y, k1) and ...and
value(x,kn) = value(y,kn) implies x = y)
```

siendo t.primaryKeys = { k_1 , …, k_n}, es decir el conjunto de atributos que formal la clave primaria.

c) La restricción de entidad establece que ningún valor de la clave primaria puede ser nulo:

```
Context t:Table inv:
t.primaryKeys -> forAll (k ¦ t.instances -> forAll (i ¦value(i,k)?null))
```

d) La restricción de integridad referencial establece que para cada clave foránea existe una tabla asociada conteniendo dicha clave. (Nota: de acuerdo con el metamodelo de UML la expresión "`t.allOppositeAssociationEnds.participant`" retorna el conjunto de todas las clases asociadas con la tabla t).

```
Context t:Table inv:
t.foreingKeys -> forAll (k ¦ t.allOppositeAssociationEnds.participant ->
exists (c ¦ c.allFeatures -> exists(f ¦f.type = k.type)))
```

6.2 Restricciones del Dominio de Aplicación

Las restricciones propias del dominio de aplicación se especifican mediante expresiones OCL asociadas al diagrama de clases. Nótese que estas expresiones predican sobre instancias del modelo (por ejemplo sobre nadadores, clubes, etc.), mientras que las expresiones de la sección 6.1 se ubican en el nivel superior predicando sobre instancias del metamodelo (por ejemplo sobre atributos, tablas, etc.). Presentamos algunos ejemplos:

a) La Posición de un nadador en un evento esta determinada por el tiempo que realice en la prueba; a menor tiempo mejor (menor) posición:

```
context c:Competition inv:
c.event -> forAll(e1, e2 ¦(e1.dateID = e2.dateID) and (e1.position <
e2.position)  implies (e1.time < e2.time) )
```

b) Todo club tiene al menos un nadador:

```
context c:Club inv:
not (c.swimmer -> isEmpty)
```

6.3. Restricciones Temporales

Definición del estereotipo <<tempTable>>: El estereotipo <<tempTable>> se aplica a instancias de la metaclase Class (y eventualmente a instancias de AssociationClass ya que esta metaclase es una especialización de Class en el metamodelo de UML). El estereotipo permite identificar tablas con valores temporales (en particular las tablas contendrán los atributos initialTime y finalTime). Este estereotipo es una especialización del estereotipo <<Table>>.

Definimos las siguientes restricciones asociadas al estereotipo (por restricciones de espacio sólo mostramos algunas a modo de ejemplo):

a) Las tablas temporales contienen dos atributos especiales para denotar el tiempo inicial y el tiempo final respectivamente:

```
Context t:TemporalTable inv:
t.allFeatures -> exists(f1,f2¦f1.name = ”initialTime” and f2.name =
”finalTime” and f1.type = Date and f2.type = Date)
```

b) Para todas las tuplas en la tabla el tiempo inicial debe ser menor o igual al tiempo final:

```
Context t:TemporalTable inv:
t.instances -> forAll(i ¦ value(i,initialTime) = value(i,finalTime))
```

7. Trabajos Relacionados

Considerando los aspectos temporales en los datawarehouse se propusieron varias soluciones. En (Bliujute et al. 1998) se presenta un esquema estrella temporal que difiere del tradicional en cuanto al tratamiento del tiempo, mientras éste toma al tiempo como una dimensión más, aquel anula la dimensión tiempo y agrega, como atributos de hecho, el tiempo inicial y final en cada una de las filas de las tablas del esquema. En (Pedersen, Jensen. 1998) se describe, entre las características que un modelo de datawarehouse debería tener, la necesidad de considerar los cambios temporales en los datos y las jerarquías no estrictas, compara los modelos existentes contra los requerimientos y observa la falta de soluciones a varios puntos citados. En (Mendelzon, Vaisman. 2000) se presenta el modelo multidimensional temporal y un lenguaje de consulta temporal, donde se agregan marcas de tiempo en las dimensiones o al nivel de instancias (o ambos) para capturar las variaciones en los atributos de las dimensiones. Relacionado con la utilización de UML como modelo de datos y a OCL como lenguaje de restricciones, en (Gogolla, Lindow. 2003) se describe mediante Meta Object Facility (MOF) la transformación del esquema entidad-interrelación al esquema relacional y utiliza OCL para establecer restricciones en el metamodelo. En (Gogolla. 2001) propone el uso de estereotipos UML para la formulación de requerimientos de un dominio específico empleando OCL para definir un significado preciso para los estereotipos introducidos. En (Behm et al. 1997) se plantean dos fases para la migración de un sistema relacional a un sistema de base de datos OO, en la primera utiliza reglas de transformación para construir un esquema OO que es semánticamente equivalente al esquema relacional, en la segunda fase ese esquema es usado para generar programas que migren los datos relacionales a una base de datos OO. En (Luján-Mora. 2002) se presenta una extensión de UML, mediante estereotipos, para representar la estructura y las propiedades dinámicas del modelado multidimensional en el nivel conceptual y utiliza XML para almacenar el modelo multidimensional. En (Gogolla et al. 2002) se estudia la sintaxis y la semántica del modelo entidad interrelación y el modelo de datos relacional y sus transformaciones. En (Gornik. 2002b) se describe el modelo multidimensional implementado en un esquema estrella y copo de nieve mediante el modelo de objetos utilizando Racional Rose. En (Gornik. 2002a) se detalla la representación, mediante UML Data Modeling Profile, de tablas, vistas e interrelaciones del modelo relacional, las restricciones se implementan mediante operaciones estereotipadas que describen el comportamiento dinámico de la base de datos.

8. Conclusión y Trabajo Futuro

En este trabajo se propone la utilización de OCL para especificar restricciones en un datawarehouse temporal. Se definen los atributos y las interrelaciones temporales y su representación en UML. Se presenta, además, una transformación del modelo multidimensional a UML y se aplican estereotipos a las clases para extender la semántica y adaptarla al dominio de las estructuras multidimensionales. Se presentan tres tipos de restricciones a las clases las propias del modelo relacional, las del dominio de aplicación y las temporales.

En cuanto a trabajos futuros, se han considerado las siguientes líneas de investigación: vinculado a las consultas temporales, la combinación en un mismo modelo de un datawarehouse con una base de datos histórica demandará de un lenguaje de consulta que explote las ventajas de tal estructura, proponiendo nuevos operadores que permitan relacionar, por ejemplo, datos históricos con los

actuales. Además, respecto a las interrelaciones n-arias temporales, nuestra trabajo analizó las interrelaciones temporales binarias. En el caso más general, las interrelaciones n-arias, la decisión de elegir el identificador único no es trivial. El análisis de la multiplicidad de cada entidad con respecto a la interrelación permitirá la elección adecuada y la posibilidad, sin perdida de información, de transformar a la interrelación n-aria en un conjunto de interrelaciones binarias y su correspondiente representación en UML.

Referencias

Agrawal R., Gupta A., Sarawagi S., Modeling Multidimensional Databases, Research Report, IBM Almaden Research Center, San Jose, California, 1995. Appeared in Proceeding. ICDE '97.

Behm, A., Geppert, A., Dittrich, K. R. "On the Migration of Relational Schemes and Data Object-Oriented Database System". In Proceedings of Re-Technologies in Information System. Klagenfurt, Austria, Dec 1997.

Bliujute R., Saltenis S., Slivinskas G., and Jensen C. S., Systematic Change Management in Dimensional Data Warehousing. in Proceedings of the Third International Baltic Workshop on Data Bases and Information Systems, Riga, Latvia, 1998.

Chaudhuri S. and Dayal U., An Overview of Data Warehousing and OLAP Technology, ACM SIGMOD Record 26(1), March 1997.

Gregersen H. and Jensen C. S., Conceptual Modeling of Time-Varying Information. TIMECENTER Technical Report TR-35. September 10, 1998.

Gregersen H., Mark L., Jensen C. S., Mapping Temporal ER Diagrams to Relational Schemas. TIMECENTER Technical Report TR-39. December 4, 1998.

Gogolla, Martin and Lindow Arne. Transforming Data Models with UML. In Borys Omelayenko and Michel Klein, editors, Knowledge Transformation for the Semantic Web, pages 18-33. IOS Press, Amsterdam, 2003.

Gogolla, Martin. Using OCL for Defining Precise, Domain-Specific UML Stereotypes. In Aybuke Aurum and Ross Jeffery, editors, Proc. 6th Australian Workshop on Requirements Engineering (AWRE'2001), pages 51-60. Centre for Advanced Software Engineering Research (CAESER), University of New South Wales, Sydney, 2001.

Gogolla, Martin, Lindow, Arne, Richters, Mark and Ziemann, Paul. Metamodel Transformation of Data Models. In Jean Bezivin and Robert France, editors, Proc. UML'2002 Workshop in Software Model Engineering (WiSME 2002).

Golfarelli M., Maio D., Rizzi S., The Dimensional Fact Model: a Conceptual Model for Data Warehouses. Invited Paper, International Journal of Cooperative Information Systems, vol 7, n.2&3, 1998.

Gornik David. UML Data Modeling Profile. Rational Software White Paper. Rational Software Corporation. 2002a

Gornik David. Data Modeling for Datawarehouse. Rational Software White Paper. Rational Software Corporation. 2002b

Jensen C. S. et al. A Consensus Glossary of Temporal Database Concepts. ACM SIGMOD Record, 23(1):52-65, March 1994.

Luján-Mora Sergio. Multidimensional Modeling using UML and XML. 12th Workshop for PhD Students in Object-Oriented Systems (PhDOOS 2002), 16th European Conference on Object-Oriented Programming (ECOOP 2002), p. 48-49: Lecture Notes in Computer Science 2548, Málaga (Spain), June 10-14, 2002.

Mc Brien P., Seltveit A. Wangler B. An Entity-Relationship Model Extended to Describe Historical Information. CISMOD pp. 244-260 July 1992.

Mendelzon A., Vaisman A., Temporal Queries in OLAP. VLDB 2000: 242-253.

Neil Carlos, Ale Juan. A Conceptual Design for Temporal Data Warehouse. 31º JAIIO. Santa Fe. Simposio Argentino de Ingeniería de Software. 2002

OCL. Object Constraint Language - version 1.5. 2002

Pedersen T. B. and Jensen C. S., Multidimensional Data Modeling for Complex Data. 1998. ICDE 1999:336-345.

Pons, Claudia, Baum,Gabriel and Felder Miguel. Foundations of Object Oriented Modeling Notations in a Dynamic Logic Framework. Fundamentals of Information Systems. Kluwer Academic Publisher. Chapter 1. 1999.

Pons, Claudia, Baum,Gabriel and Felder Miguel. Formal Foundations of Object Oriented Modeling Notations. 3th International Conference on Formal Engineering Methods, IEEE ICFEM 2000. UK.

Richters, Mark and Gogolla, Martin. Validating UML Models and OCL Constraints. In Andy Evans and Stuart Kent, editors, Proc. 3rd Int. Conf. Unified Modeling Language (UML'2000), pages 265-277. Springer, Berlin, LNCS 1939, 2000.

UML. The Unified Modeling Language Specification – version 1.3. 2001

Vassilliadis P, Sellis T. A Survey on Logical Models for OLAP Databases. SIGMOD Record 28(4), pp. 64-69, December 1999.