Métodos y Herramientas para el Desarrollo de Aplicaciones

The Distributed Group SEVILLE

Actas del Taller de Trabajo ZOCO

R. Corchuelo, D. Ruiz y J.L. Arjona
(Editores)

Zoco es un taller de trabajo organizado periódicamente por el Grupo de Sistemas Distribuidos de la Universidad de Sevilla con el propósito de dar cabida en él a investigadores y profesionales que trabajan en métodos y herramientas para el desarrollo de aplicaciones, con un marcado énfasis en la Web.

En este libro se recogen las actas de dicho taller, que fue celebrado en el contexto de las Jornadas de Ingeniería del Software y Bases de Datos en Octubre de 2006 en Sitges. Podrá encontrar información adicional sobre estas y otras ediciones en la dirección http://www.tdg-seville.info/cfp/Zoco.

JORNADAS DE
Ingeniería del Software y Bases de Datos
Sitges, 3 al 6 de Octubre de 2006
JISBD 2006
MINISTERIO DE EDUCACIÓN Y CIENCIA
mec.es

Métodos y Herramientas para el Desarrollo de Aplicaciones – R. Corchuelo, D. Ruiz, J.L. Arjona (Editores)

Métodos y Herramientas para el Desarrollo de Aplicaciones

Actas del Taller de Trabajo ZOCO

Rafael Corchuelo, David Ruiz y José Luis Arjona
(Editores)

Métodos y Herramientas para el Desarrollo de Aplicaciones

Actas del Taller de Trabajo ZOCO

Rafael Corchuelo, David Ruiz, José L. Arjona

Publicado por The Distributed Group
ETSI Informática

ISBN: 978-84-690-5792-6

Imprime Artes Gráficas Domínguez, S.L.L.
Impreso en España

Financiación: Proyecto AgilWeb (TIC2003-02737-C02-01) y Acción Complementaria Zoco (TIN2005-24794-E)

Índice general

Índice de Figuras

Índice de Tablas

Capítulo 5

Primer paso para la obtención de servicios en bases de datos relacionales mediante patrones y MDA

Ignacio García-Rodríguez, Macario Polo, Mario Piattini
Escuela Superior de Informática. Universidad de Castilla–La Mancha
Paseo de la Universidad, 4. Ciudad Real 13071
(Ignacio.GRodriguez,Macario.Polo,Mario.Piattini)@uclm.es

SOA surge como una iniciativa para que las organizaciones abran sus negocios a los nuevos horizontes y escenarios propiciados por la evolución del software. Esta evolución pasa por la distribución, la integración de sistemas heterogéneos y la capacidad para la adaptación al cambio. MDA se vislumbra como una solución eficiente para esta adaptación de los sistemas de información a SOA. Este artículo aborda una pequeña parte de esa orientación del software a servicios, estudiando cómo mediante un enfoque orientado a modelos, se pueden extraer posibles servicios a partir de una base de datos, para ser finalmente ofrecidos mediante Servicios Web.

5.1. Introducción

Los Servicios Web y la Arquitectura Orientada a Servicios (SOA) son tecnologías que están experimentando una gran aceptación lo que ha promovido su desarrollo. En concreto los Servicios Web han facilitado que los sistemas heredados (que suelen constituir el núcleo de los sistemas de información de las organizaciones) puedan seguir funcionando y mejorar su eficiencia [95].

Entre otros componentes de esos sistemas heredados, es frecuente encontrar bases de datos, que se presentan como el corazón de los mismos, manteniendo toda la información que estos necesitan. Y como parte de esos sistemas heredados, las bases de datos también deben ser tenidas en cuenta a la hora de migrar los sistemas hacia una arquitectura orientada a servicios [36]. Dentro de éstas, cabe destacar las bases de datos relacionales, concretamente aquellas basadas en el estándar SQL-92 [45]. A pesar de que se han desarrollado múltiples versiones del estándar SQL (como SQL-99 y SQL-2003), muchos sistemas están funcionando todavía con bases de datos relacionales [13] basadas en el estándar SQL-92. Por esta razón, las bases de datos SQL-92 constituyen auténticos sistemas heredados susceptibles de ser integradas como parte fundamental de los sistemas de información de las organizaciones.

Una de las herramientas que están surgiendo con el MDA [66] para dar soporte a tareas tan ambiciosas como la migración de sistemas heredados hacia SOA es el ADM (Architecture-Driven Modernization) [21]. ADM propone el uso de un enfoque orientado a modelos para renovar y migrar aquellos sistemas que, aun siendo antiguos y desarrollados bajo tecnologías obsoletas, siguen siendo vitales para las organizaciones. En [36] se presenta una metodología para la obtención de servicios a partir de bases de datos SQL-92. Estos servicios son posteriormente ofrecidos mediante Servicios Web. Esta metodología, basada en el enfoque MDA y ADM, ayuda a la integración de bases de datos en los entornos SOA, dada la importancia que éstas en los sistemas de información actuales [13, 14].

En este artículo se presentan varios de patrones para explorar bases de datos SQL-92 en busca de posibles servicios. Tanto los patrones de búsqueda como la base de datos están representados en términos de modelos, por los que esta búsqueda puede verse como un Model-Driven Pattern Matching (MDPEM, en adelante), concepto introducido en [36] o, dicho de otro modo, como una búsqueda de patrones dirigida por modelos. Dichos patrones se integran dentro de la metodología mencionada, constituyendo la búsqueda de servicios como una de las etapas principales de la misma.

El artículo se organiza de la siguiente manera: la Sección §5.2 ofrece algunas referencias sobre el uso de los patrones en la Ingeniería del Software; la

Sección §5.3 profundiza en el concepto de Model-Driven Pattern Matching, necesario antes de abordar los patrones; la Sección §5.4 presenta brevemente algunos patrones básicos para la búsqueda de servicios; la Sección §5.5 ofrece algunas conclusiones y las posibles líneas de trabajo futuro.

5.2. Estado del arte

Tradicionalmente los patrones han sido utilizados con múltiples objetivos, entre los que podemos mencionar la migración de sistemas, como ocurre en [17], donde se emplea el patrón de diseño MVC para la migración de un sistema heredado basado en COBOL hacia una aplicación web basada en Java y páginas web. [38] también trata de refactorizar sistemas heredados mediante un lenguaje precisamente basado en lenguajes arquitectónicos.

En otros trabajos, [93, 94] los patrones han sido empleados para conectar dos paradigmas como el mundo de los objetos y las bases de datos relacionales, estableciendo cómo mapear objetos a tablas en una base de datos y gestionar su persistencia. Relacionado con las bases de datos, también cabe mencionar a [72], donde se desarrolla una nueva versión del patrón CRUD. RCRUD se basa en el análisis, mediante reflexión, de objetos Java para la creación de código para la gestión de su persistencia.

En [23] se emplea el patrón adaptador para construir wrappers capaces de aislar un sistema heredado y ofrecer su funcionalidad como un servicio. En [71], se baja el nivel de abstracción, utilizando los patrones para analizar a nivel léxico y sintáctico el código fuente de una aplicación, permitiendo así crear representaciones abstractas que mejoren la comprensión del ingeniero sobre el sistema. [79, 80] persiguen también la mejora de la comprensión del usuario respecto del sistema, donde los patrones representan consultas lanzadas por el propio usuario. El matching se lleva a cabo contra un diagrama entidad-relación que representa el sistema en sí. Estos tratan de obtener una visión de la arquitectura general del sistema, tal y como se hace también en [84].

5.3. Model-Driven Pattern Matching

De manera introductoria a los patrones propuestos en este artículo, se va a esbozar el concepto de MDPEM mencionado en la sección de introducción. Este concepto se introducirá mediante un ejemplo que dará entender intuitivamente cómo se realiza la búsqueda mediante patrones a nivel de modelo.

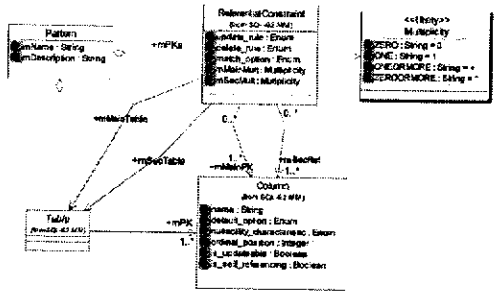


Figura 5.1: *Metamodelo de patrón para la búsqueda en un modelo SQL-92.*

Supongamos en primer lugar un modelo M (conforme a un metamodelo para base de datos), que describe una base de datos relacional en términos de tablas, columnas, claves ajenas, dominios, etc. y un modelo P, constituido por un conjunto de clases y relaciones entre ellas, que puede ser visto como un fragmento de una base de datos. La clave está en que el modelo P puede ser utilizado como patrón para encontrar en M conjuntos de clases (representados por tablas) que sigan la misma estructura que el modelo P, esto es, en términos de número de clases y relaciones entre éstas.

Precisamente esa búsqueda de ocurrencias de un modelo en otro se corresponde con el concepto de MDPEM. Esta idea se esboza en la especificación de MDA y en la especificación QVT [67]. Según [76], la búsqueda de patrones consiste en que una expresión concisa de restricciones complejas se contraste con un dato de entrada, de forma que aquellas partes que coincidan con el patrón sean seleccionadas y devueltas. En el caso del MDPEM, el tipo de dato de entrada sería un modelo, al igual que el patrón de búsqueda.

En este artículo, el modelo que representa el patrón es una descripción genérica de un servicio que se utilizará para realizar el proceso de matching contra el modelo que represente la base de datos. A tal fin, se proponen los metamodelos de patrón de la figura §5.1. A diferencia de otros metamodelos [69], el propuesto en figura §5.1 está acotado al contexto en el que se desenvuelve la metodología, y permite hacer el matching sobre un modelo que represente una base de datos en SQL-92.

Poder especificar algunos patrones en el contexto de la estructura de una base de datos, permitiría encontrar grupos de tablas que cumplan con él. Además, cada patrón tendría asociadas un conjunto de operaciones genéricas (como veremos más adelante), de forma que cuando se encuentren las ocurrencias de ese patrón en la base de datos, se puedan generar las operaciones asociadas al patrón, pero involucrando a tablas reales de la base de datos, y constituyendo así una funcionalidad susceptible de ser ofrecida como parte de un Servicio Web.

A modo de ejemplo, se va a ilustrar el concepto de MDPEM mediante la figura §5.2, donde se muestra un modelo que representa la información bancaria de una persona (dado que la metodología mencionada sigue la filosofía MDA, este modelo constituiría el PSM del proceso, ya que sería la representación de la base de datos, que es el sistema heredado).

Como se ve en el modelo de entrada, un cliente tiene cuentas, las cuentas tienen asociados préstamos, cada cuenta tiene asociadas tarjetas que pueden ser de débito o crédito, además, cada tarjeta sea del tipo que sea, tiene asociados una serie de movimientos. El modelo de entrada representaría una instancia del metamodelo de la base de datos (propuesto en [36]), construido a partir de los metadatos del diccionario de la base de datos. El patrón de consulta, que sería una instancia del metamodelo de la figura §5.1, representa cuatro clases (A,B,C y D) relacionadas entre si mediante tres claves ajenas (Fk_{ba}, Fk_{cb} y Fk_{db}). El MDPEM tomaría el modelo de entrada, y con el patrón de consulta buscaría las vistas del modelo de la base de datos que cumplieran las restricciones del patrón, es decir, representa de manera genérica la estructura que se desea encontrar en el modelo de entrada.

El matching del patrón de consulta sobre el modelo de entrada, puede verse como una pseudoconsulta SQL realizada sobre una base de datos, donde el patrón jugaría el rol de las condiciones a cumplir y los elementos del esquema que se desea recuperar, y el modelo de entrada actuaría como la base de datos de donde se desea recuperar la información que cumpla con las condiciones establecidas patrón especificado. La pseudoconsulta correspondiente al patrón y al modelo de la figura §5.2 sería la siguiente:

SELECT $A, B, C, D, Fk_{ba}, Fk_{cb}, Fk_{db}$ FROM Modelo_de_Entrada
WHERE $Fk_{ba}(A, B)$ AND $Fk_{cb}(B, C)$ AND $Fk_{db}(B, D)$

Dicha consulta se interpretaría así: seleccionar las tablas A,B,C y D, y las claves ajenas Fk_{ba}, Fk_{cb} y Fk_{db} del modelo Modelo_de_Entrada cuando cumplan la condición de que las clases A y B estén vinculadas mediante Fk_{ba}, la clases B y C estén vinculadas mediante Fk_{cb}, y que las clases B y D estén vinculadas mediante Fk_{db}. Al ejecutarse el matching se recuperarían todos los grupos de tablas y claves ajenas que cumplieran las condiciones expuestas. Concretamente para el modelo de entrada y el patrón propuestos en la figura §5.2, los matchings o coincidencias serían cuatro. Cada una de estos matchings

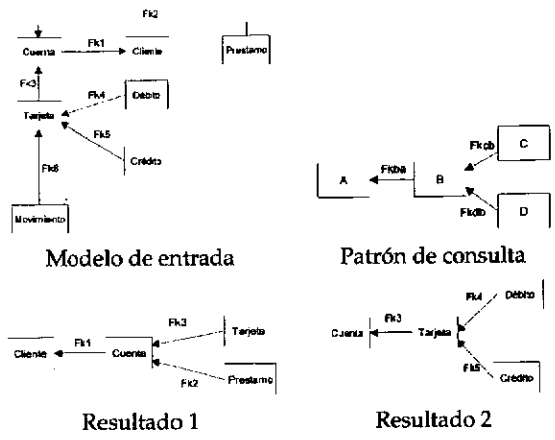


Figura 5.2: *Modelo, patrón y resultados del matching.*

estarían compuestos por cuatro tablas (representadas a nivel conceptual mediante la instancia del metamodelo) que coincidirían con las tablas A, B, C y D, y tres claves ajenas, que coincidirían con Fk_{ba}, Fk_{cb} y Fk_{db}. La tabla §5.1 resume algunos de los posibles resultados que se obtienen de la aplicación del MDPEM utilizando el patrón de la figura §5.2 (dcha) sobre el modelo que representa de la base de datos, a la figura §5.2 (izda).

5.4. Patrones de descubrimiento básicos

Los patrones siguientes son un conjunto básico que puede ser extendido con nuevos patrones adaptados al metamodelo de la figura §5.1.

5.4.1. Patrón Clave Ajena (CA)

Una clave ajena puede resultar muy útil para encontrar funcionalidad. Por ejemplo, dadas las tablas A y B y la clave ajena Fk de la figura §5.3, podemos encontrarnos con los siguientes casos:

Patrón	Resultado 1	Resultado 2
A	Cliente	Cuenta
B	Cuenta	Tarjeta
C	Préstamo	Débito
D	Tarjeta	Crédito
Fk_{ba}	Fk_1	Fk_3
Fk_{cb}	Fk_2	Fk_4
Fk_{db}	Fk_3	Fk_5
Fk_{ba} (A,B)	Fk_1 (Cliente, Cuenta)	Fk_3(Cuenta, Tarjeta)
Fk_{cb} (B,C)	Fk_2 (Cuenta, Préstamo)	Fk_4(Tarjeta, Débito)
Fk_{db} (B,D)	Fk_3 (Cuenta, Tarjeta)	Fk_5(Tarjeta, Crédito)

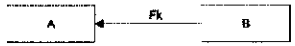
Tabla 5.1: *Equivalencias entre el patrón de entrada y los matchings.*

- $Pk_a = colRef_a$ y $Pk_b = colRef_b$: Las columnas que establecen la clave ajena en B hacen referencia a la clave primaria de A. La relación entre A y B es 1:1, y podría interpretarse como una relación de herencia en la orientación a objetos, donde B representaría una entidad que heredaría las características de A. Las operaciones que se obtendrían sería del tipo $getAbyB(Pk_b)$ y $getBbyA(Pk_a)$.
- $Pk_a = colRef_b$ y $colRef_b \subset Pk_b$: Esta relación puede interpretarse como una relación de herencia donde B hereda de A, y además B añade más características. Es decir, $Pk_b = Pk_a + X$, donde X son las características que añade B, que se traducirían a columnas de la clave primaria de B. Se generarían las siguientes operaciones: $getAbyB(Pk_b)$, $getBbyA(Pk_a)$ y $getAbyB(X)$.
- $\cap(colRef_b, Pk_b) = \emptyset$): En este caso, la relación es de 1:N

Las operaciones de los casos expuestos, permitirán obtener registros de la tabla referenciada a partir de la tabla referenciante y, al contrario, un conjunto de registros de la tabla referenciante a partir de la tabla referenciada.

5.4.2. Patrón Doble Clave Ajena (DCA)

En este caso, la estructura para la búsqueda de operaciones será algo más compleja que la anterior. Concretamente se buscan tres tablas, de las cuales dos de ellas son referenciadas por la tercera, que es inicio de dos claves ajenas (figura §5.4).



Pk_a = Clave primaria de A; Pk_b = Clave primaria de B; $colRef_b$ = Conjunto de columnas de B que referencia a Pk_a; $colRef_a$ = Conjunto de columnas de A referenciadas por B en la Fk.

Figura 5.3: *Relación entre tablas A y B por una clave ajena* Fk.

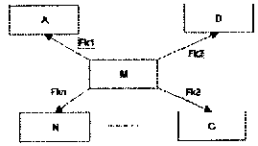
Pk_a = Clave primaria de A; Pk_b = Clave primaria de B; Pk_m = Clave primaria de M; $colRef_{ma}$ = Conjunto de columnas de M que referencia a A; $colRef_{mb}$ = Conjunto de columnas de M que referencia a B.

Figura 5.4: *Relación entre las tablas A, B y M con las claves ajenas* Fk_1 *y* Fk_2.

Este patrón ofrece posibilidades especiales a la hora de recorrer colecciones de registros en las tablas vinculadas. En función de las columnas de la tabla M, podemos encontrarnos con los siguientes casos:

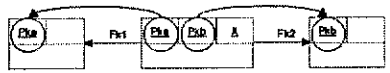
- Pk_m coincide con $colRef_{ma} + colRef_{mb}$ y $Pk_m = Pk_a + Pk_b$: La clave primaria de la M constituye las claves ajenas, y además, la clave primaria de M es igual a la unión de las claves primarias de A y B. De acuerdo a la figura §5.4, algunas de las operaciones generadas serían: `getAbyB` y `getBbyA`.

- $(colRef_{ma} + colRef_{mb}) \supset Pk_m$ y $colRef_{ma} = Pk_a$ y $colRef_{mb} = Pk_b$: En este caso, la unión de las claves ajenas ($colRef_{ma}$ y $colRef_{mb}$) constituyen un subconjunto de la clave primaria (Pk_m), y además coincide con las claves primarias de A y B. Sin embargo, la clave primaria de M tiene columnas adicionales a las que constituyen las claves ajenas. Podríamos mencionar las siguientes operaciones para este caso: `getAbyB`, `getBbyA`, `getAbyX`, `getBbyX`, `getABbyX`, etc.

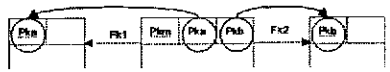


Pk_a = Clave primaria de A; Pk_b = Clave primaria de B; Pk_c = Clave primaria de C; Pk_n = Clave primaria de N; Pk_m = Clave primaria de M; $colRef_{ma}$ = Conjunto de columnas de M que referencia a A; $colRef_{mb}$ = Conjunto de columnas de M que referencia a B; $colRef_{mc}$ = Conjunto de columnas de M que referencia a C; $colRef_{mn}$ = Conjunto de columnas de M que referencia a N.

Figura 5.5: *Interrelación n-aria.*



- $\cap(Pk_m, colRef_{ma}) = \emptyset$ y $\cap(Pk_m, colRef_{mb}) = \emptyset$ y $colRef_{ma} = Pk_a$ y $colRef_{mb} = Pk_b$: La clave primaria de M no está compuesta por ninguno de los conjuntos de columnas que forman las claves ajenas ($colRef_{ma}$ y $colRef_{mb}$), siendo disjunto entre sí, y referencian respectivamente a las claves primarias de las tablas A y B (Pk_a y Pk_b). Para este caso, algunas de las operaciones serían: `getAbyB`, `getBbyA`, `getABbyM`, `getAbyBM`, `getBbyAM`, etc.



5.4.3. Patrón Clave Ajena n-aria (CANA)

Este patrón puede considerarse una extensión del patrón Doble Clave Ajena, ya que se considera la posibilidad de encontrar estructuras similares a la ya citada, pero con la salvedad de que ahora nos podemos encontrar con n claves ajenas, con n mayor que 2 (ver figura §5.5).

En función de cómo estén establecidas las claves ajenas y las claves primarias en la tabla M, podemos distinguir los mismos casos que en el patrón DCA, pero extendiendo las restricciones a un número n>2:

- La unión de claves ajenas de la tabla M hacia las n tablas referenciadas forman la clave primaria de M, además, ninguna de las claves ajenas comparten ninguna columna. Las condiciones para el cumplimiento del patrón pueden resumirse de la siguiente manera:

$$(\forall \mathrm{colRef}_{mi} \in M, \mathrm{colRef}_{mi} = \mathrm{Pk}_i \wedge \mathrm{colRef}_{mi} \in \mathrm{Pk}_i) \wedge$$
$$(\forall i, j \in \{1..n\}, \cap(\mathrm{colRef}_{mi}, \mathrm{colRef}_{mj}) = \emptyset)$$

- En este caso, la clave primaria además de estar compuesta por la unión disjunta de las claves ajenas, tiene atributos adicionales, denotados por el conjunto X. Las restricciones de este patrón pueden formalizarse de la siguiente manera:

$$(\mathrm{Pk}_m = \cup(\mathrm{colRef}_i, X), i \in \{1..n\}) \wedge$$
$$(\forall i, j \in \{1..n\}, \cup(\mathrm{colRef}_i, \mathrm{colRef}_j) = \emptyset)$$

- Las claves ajenas no forman parte de la clave primaria de la tabla M, y además son disjuntas entre si, es decir, que no comparten ninguna columna. La expresión formal de dichas restricciones se pueden expresar de la siguiente manera:

$$(\forall i, j \in \{1..n\}, \cap(\mathrm{colRef}_{mi}, \mathrm{colRef}mj) = \emptyset) \wedge$$
$$(\forall i \in \{1..n\}, \cap(\mathrm{colRef}_{mi}, \mathrm{colRef}_m) = \emptyset)$$

De esta forma, en función del matching del patrón que se detecte en el esquema de la base de datos, podrán generarse un conjunto u otro de operaciones:

Caso 1. Dadas las tablas A, B y C (ver figura §5.6), formando una relación ternaria mediante la tabla M, y cumpliendo las premisas de este caso, las operaciones que nos permitirían navegar entre las tuplas de las tablas serían: getABbyC, getACbyB, getBDbyA, getAbyBC, getBbyAC, y getCbyAB. Donde las tres primeras tomarían como argumento un valor fijo para la consulta (el valor de las claves primarias de C, B o A respectivamente), y las tres siguientes tomarían como argumentos los valores las claves primarias de AB, AC, o BC. En total sumarían 6 operaciones generadas a partir de un matching del patrón en la base de datos.

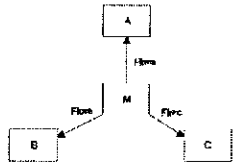


Figura 5.6: *Relación n-aria de con n = 3.*

Caso 2. El segundo caso puede considerarse una especialización del primero, donde la clave primaria de M, además de estar compuesta por la suma de las claves ajenas a las n tablas que forman la relación n-aria, existe un conjunto de atributos adicionales denotado con la letra X. Dada la relación de la figura §5.6, las operaciones que podríamos deducir a partir de las claves ajenas serían: getABCbyX, getXbyABC, getABbyCX, getACbyBX, getBCbyAX, getAbyBCX, getBbyACX y getBCbyA.

Caso 3. En este tercer caso, las claves ajenas no forman parte de la clave primaria de M, y además son disjuntas entre sí. Es posible que la existencia de otra clave primaria distinta a la unión de las claves ajenas sea debido a un mal diseño de la base de datos (por ejemplo cuando la clave primaria es un campo autonumérico). Sin embargo puede presentarse el caso en un sistema heredado y es necesario tenerlo en cuenta. Las operaciones que pueden generarse ahora responden exactamente al mismo criterio seguido en el caso 1, y además, podemos añadir otras dos: getABCbyPk$_m$ y getPk$_m$byABC. Con lo que el número de operaciones que se obtienen responden a la misma forma que en el caso 2. Si Pk$_m$ estuviera compuesta por un campo autonumérico, las operaciones getABCbyPk$_m$ y getPk$_m$byABC no tendrían mucho sentido. Sin embargo si la clave primaria estuviera compuesta por otro tipo de columnas es posible que estas dos operaciones resultaran de alguna utilidad.

5.5. Conclusiones y trabajo futuro

En este artículo se ofrece una nueva visión de la búsqueda mediante patrones, basando la búsqueda en el enfoque MDA para facilitar la labor de la integración de bases de datos relacionales SQL-92 en entornos SOA. Se ha descrito, de manera general, en qué consiste el proceso de búsqueda de ocurrencias (matchings) de un patrón, representando un modelo en un esquema de

base de datos que está a su vez representado por otro modelo, denominado MDPEM. Estos patrones se integran como una parte de una metodología para la extracción de servicios a partir de bases de datos relacionales. Esta metodología utiliza QVT [67] como lenguaje para implementación de todas las transformaciones entre modelos y, tal y como se esboza en este artículo, para la búsqueda de servicios en la base de datos mediante patrones. Se ha presentado un conjunto reducido de patrones para la búsqueda de funcionalidades. Estos patrones resultan útiles para explorar la estructura de una base de datos SQL-92. Sin embargo, la experimentación mediante casos de estudio puede ayudar a revelar patrones más complejos que los aquí citados. Como parte de la metodología citada, la generación de código para la implementación de los servicios, y la propia generación de los Servicios Web para ofertar vía web estas operaciones forma parte del trabajo en curso.

Agradecimientos

Este trabajo ha sido parcialmente financiado por el proyecto MÁS, Ministerio de Ciencia y Tecnología/FEDER, TIC2003-02737-C02-02; el proyecto ENIGMAS, Plan Regional de Investigación Científica, Desarrollo Tecnológico e Innovación, Junta de Comunidades de Castilla-La Mancha, PIB-05-058, el proyecto FAMOSO, parcialmente financiado por el Ministerio de Industria, Turismo y Comercio, FIT-340000-2006-67 Plan Nacional de Investigación Científica, Desarrollo e Innovación Tecnológica 2004-2007 y el Fondo Europeo de Desarrollo Regional (FEDER), Unión Europea, y el proyecto MECENAS, Junta de Comunidades de Castilla-La Mancha, Consejería de Educación y Ciencia, PBI06-0024.