Universidad Central "Marta Abreu" de Las Villas
Facultad de Matemática, Física y Computación
Licenciatura en Ciencia de la Computación

Trabajo de Diploma

Integración de esquemas XML heterogéneos

Autor: Yanet Pedraza Costa

Tutores: MSc. Darien Rosa Paz

Santa Clara

2011

DICTAMEN

El que suscribe _____ ,
hago constar que el trabajo titulado _____
_____fue realizado en la Universidad Central “Marta Abreu” de Las Villas como parte de la culminación de los estudios de la especialidad de _____, autorizando a que el mismo sea utilizado por la institución, para los fines que estime conveniente, tanto de forma parcial como total y que además no podrá ser presentado en eventos ni publicado sin la autorización de la Universidad.

_____

Firma del autor

Los abajo firmantes, certificamos que el presente trabajo ha sido realizado según acuerdos de la dirección de nuestro centro y el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

_____ _____

Firma del tutor Firma del jefe del Laboratorio

_____

Fecha

RESUMEN

La conciliación de múltiples fuentes de datos heterogéneas ha recibido especial atención en el desarrollo de bases de datos. El rápido y continuo crecimiento actual de las grandes organizaciones empresariales impone una necesidad creciente en la integración y la distribución de grandes cantidades de datos, procedentes de varias fuentes de datos heterogéneos y distribuidos. Con el impresionante crecimiento de Internet y la propuesta consecuente del estándar XML para el intercambio de datos en la Web, la necesidad de integrar una amplia variedad de formatos y datos en la Web ha renovado el interés en la integración de datos semiestructurados. Se han desarrollado varios sistemas para la integración de datos semiestructurados basados en arquitecturas multicapas.

El sistema de integración de datos Automed ha sido extendido para dar soporte a la integración de datos XML y propone un nuevo enfoque para llevar a cabo este proceso.

Esta investigación se basa en la metodología propuesta por el sistema Automed para desarrollar una herramienta que integre documentos XML heterogéneos en un esquema global, con el fin de que pueda ser integrada posteriormente a un Sistema de Recuperación de Información sobre archivos XML con estas características.

ABSTRACT

Reconciling multiple heterogeneous data sources has been a major focus of database research. Today's fast and continuous growth of large business organizations enforces an increasing need in integrating and sharing large amounts of data coming from a number of heterogeneous and distributed data sources. With the impressive growth of Internet and the consequent proposal of the XML standard for the exchange of data on the Web, the need to integrate a wide variety of formats and data on the Web has renewed interest in semi structured data integration. Several systems have been built to integrate semi structured data, based on multi-tier architectures.

Automed data integration system has been extended to support the integration of heterogeneous XML documents and proposes a new approach to perform this process.

This research is based on the Automed's methodology for developing a tool able to integrate XML heterogeneous data sources into a global schema, so it can be integrated afterwards to an Information Retrieval System over XML documents with these features.

ÍNDICE

INTRODUCCIÓN

Desde el surgimiento de las bases de datos distribuidas, heterogéneas y multibases de datos, el acceso e integración de la información heterogénea ha sido un problema para el cual no existe una solución completa. Técnicas como esquemas globales, integración de esquemas, manejo de esquemas múltiples, y transacciones globales, han producido pasos significativos pero nunca han alcanzado el estado de madurez necesario para un uso a gran escala. En la actualidad, el problema es aún más complicado dado que los repositorios existen en varios formatos (HTML, XML, bases de datos relacionales, bases de datos en la Web con interfaces de consulta, etc.) y esquemas, y tanto el contenido como su estructura cambian constantemente. Este fenómeno ha conducido a la proliferación de sistemas de integración de datos basados en múltiples arquitecturas tales como almacenes de datos y sistemas mediadores. Sin embargo, a pesar del hecho de que una cantidad creciente de datos en la Web se ha estado reformando como archivos XML, los sistemas que tratan específicamente con la integración de datos XML solo están empezando a emerger.

El problema de la integración de datos ha sido abarcado en dos contextos principales: la integración tradicional de esquemas en bases de datos heterogéneas y la integración de datos semiestructurados.

Tradicionalmente, la integración de datos opera al nivel de esquemas. La estructura de cada fuente de datos es caracterizada por un esquema y se le suministra al usuario un esquema integrador (Abiteboul et al., 2005).

Ante un problema donde los documentos XML presentan estructuras heterogéneas se hace necesaria la integración de las mismas en una estructura unificada. El resultado de la integración sería un DTD (W3C, 1998) unificado y un mecanismo de enlace de cada una de las fuentes individuales a este DTD. De esta forma una consulta puede ser expresada en términos del DTD unificado y traducida automáticamente a cada una de las fuentes de datos (Szlávik and Rölleke, 2005).

Diversos autores han abordado el tema de la integración de esquemas en documentos XML con diferentes estructuras (Abiteboul et al., 2005, Lehtonen, 2005, Sauvagnat and Boughanem, 2005, Azevedo et al., 2006, Larson, 2007, Frommholz and Larson, 2007, Passi et al., 2002). Sin embargo algunas de las

herramientas obtenidas como resultado de sus investigaciones no son de carácter público.

La Recuperación de Información (RI) en XML (RI-XML) combina características de la RI tradicional y la RI sobre bases de datos. Debido a que los documentos XML separan el contenido de su estructura, los SRI-XML son capaces de retornar como respuesta a una consulta resultados más específicos dentro de un documento y no el documento en su totalidad.

En entornos reales de la RI-XML requiere tratar con documentos XML que presentan estructuras heterogéneas (diferentes DTDs) pues una misma base de datos documental puede almacenar documentos de diferentes propósitos, autores y fuentes (Szlávik and Rölleke, 2005).

Atendiendo a lo expuesto anteriormente se percibe como **problema de investigación** la necesidad de una herramienta que obtenga un esquema global a partir de esquemas XML (W3C, 2001) heterogéneos que pueda ser integrada posteriormente en un SRI sobre archivos XML con estas características.

De esta forma, el **objetivo general** del presente trabajo es obtener una herramienta que integre esquemas XML heterogéneos en un esquema global para que forme parte de un SRI XML que tenga como colección documental documentos XML con estructuras diferentes.

Para dar cumplimiento a este objetivo se plantean los siguientes **objetivos específicos**:

1. Hacer un estudio de los sistemas de integración de datos que se reportan en la bibliografía y en especial de los sistemas de integración de archivos XML.
2. Desarrollar una herramienta que integre esquemas XML heterogéneos en un esquema global.

Preguntas de investigación:

1. ¿Cuáles de los técnicas abordadas en la literatura para la integración de esquemas heterogéneos emplear en la herramienta?
2. ¿Qué técnicas de expansión de consultas se podrían aplicar en la implementación de la herramienta?

Justificación de la investigación:

En el moderno escenario de sistemas de información globales se evidencia un incremento del número de fuentes de datos estructuradas (como bases de datos

relacionales y repositorios de documentos en XML) que requieren una interfaz de consulta uniforme para acceder a los datos distribuidos y mediar entre las representaciones heterogéneas de los datos en las diversas fuentes.

Debido a la amplia aceptación del lenguaje XML como estándar para el modelado, almacenamiento e intercambio de datos estructurados, su potencialidad para mezclar información estructurada y no estructurada, así como a las ventajas que ofrece para la RI, este formato de datos constituye el más popular para representar documentos estructurados que formarán la colección documental de un SRI estructurado.

Los resultados de esta investigación se podrán aplicar en múltiples escenarios como en la medicina, bibliotecas o repositorios científicos, con un impacto positivo desde el punto de vista social, educacional y económico, pues incorporará a los SRI-XML existentes la posibilidad de consultar una colección documental formada por documentos XML con esquemas heterogéneos.

Viabilidad de la investigación:

El volumen de trabajo que implica este proyecto se puede ajustar a un período de seis meses; teniendo en cuenta que existen los equipos y recursos necesarios para su desarrollo.

Este documento está organizado de la siguiente manera. En el primer capítulo se hace un repaso de los conceptos generales de la integración de datos, destacándose las técnicas y enfoques para este proceso abordados en la literatura. También se hace un sondeo de los sistemas de integración de datos que se han desarrollado, haciendo énfasis en los sistemas de integración de datos XML. En el segundo capítulo se analiza detalladamente el proceso de integración según la metodología propuesta por Automed y se describen las API proporcionadas por este sistema que fueron utilizadas para el desarrollo de la herramienta. En el tercer capítulo se presentan los aspectos a tener en cuenta para la utilización de la herramienta, los resultados de las pruebas efectuadas y se analizan sus vulnerabilidades.

CAPÍTULO I. **LA INTEGRACIÓN DE DATOS**

En este capítulo se presenta un marco teórico que abarca los aspectos principales relacionados con la integración de datos. Se presentan las técnicas para la correspondencia de esquemas y se analizan los diferentes enfoques que se utilizan en el diseño de los sistemas de integración datos. Se expone una panorámica del estado del arte en la integración de información heterogénea haciendo énfasis en la integración de datos en formato XML.

1.1 Visión general sobre cuestiones de integración de datos

La integración de datos consiste en combinar datos alojados en diferentes fuentes, y proveer al usuario con una vista unificada de estos datos (Ullman, 1997, Halevy, 2001, Hull, 1997). El problema de diseñar sistemas de integración de datos es importante en aplicaciones actuales del mundo real, y está caracterizado por un número de cuestiones que son interesantes desde un punto de vista teórico.

Los sistemas de integración de datos en los que se centra la atención en este trabajo están caracterizados por una arquitectura basada en la obtención de un esquema global a partir de un conjunto de fuentes de datos. A estos sistemas se les denomina sistemas de integración de datos jerárquicos. Las fuentes de datos contienen los datos reales, mientras que el esquema global proporciona una vista virtual, integrada y unificada de las fuentes subyacentes. El modelado de la relación entre las fuentes y el esquema global es por tanto un aspecto crucial. Varios enfoques han sido planteados para este propósito. El primer enfoque, nombrado *global-as-view,* requiere que el esquema global sea expresado en términos de las fuentes de datos. El segundo enfoque, llamado *local-as-view*, requiere que el esquema global sea especificado independientemente de las fuentes, y la relación entre el esquema global y las fuentes se establecen definiendo cada fuente como una vista sobre el esquema global. El tercer enfoque denominado *both-as-view* se basa en secuencias reversibles de transformaciones primitivas de esquema. Más adelante se discutirán las características de estos mecanismos de modelado.

Independientemente del método usado para la especificación de la correspondencia entre el esquema global y las fuentes, un servicio básico proporcionado por el sistema de integración de datos es dar respuesta a consultas planteadas en términos del esquema global. El procesamiento de consultas en la

integración de datos requiere un paso de reformulación: la consulta sobre el esquema global tiene que ser reformulada en términos de un conjunto de consultas sobre las fuentes.

A continuación se relacionan aspectos relevantes a tener en cuenta a la hora de diseñar un sistema de integración de datos.

• El problema de la creación de un marco lógico para la integración de datos, especificando el esquema a ser accedido, las fuentes de datos y la relación existente entre ellos (el *mapeo*), y mediante la asignación de una semántica a la especificación. Una cuestión que está relacionada con el procesamiento de consultas, ya que cualquiera sea el enfoque que se esté usando, el procesamiento de consulta tiene que ser sólido y completo con respecto a la semántica definida para consultas sobre el sistema (Lenzerini, 2002).

• Gestión de modelo (Bernstein, 2003), cuyo objetivo es implementar una infraestructura genérica para gestionar esquemas, que es aplicable a todos los posibles modelos de datos subyacentes.

• Correspondencia de esquemas (Ogilvie and Callan, 2006), cuyo objetivo es proporcionar a la aplicación una *herramienta de mapeo* semiautomática que dado dos esquemas como entrada, produce un mapeo entre elementos de los dos esquemas que corresponden semánticamente el uno al otro.

• La construcción automática de las *capas,* que son responsables de presentar los datos en las fuentes en una forma adecuada para su uso en el sistema de integración de datos.

• El problema de incorporar en el marco de integración de datos nociones de calidad (calidad de datos, calidad de las respuestas) y *limpieza de datos* (Bouzeghoub and Lenzerini, 2001).

• Optimizaciones de consultas sobre un sistema de integración de datos (Adali et al., 1996).

1.2 Sistemas de integración de datos jerárquicos

Los sistemas de integración de datos jerárquicos pueden caracterizarse por una terna $I = (G, S, M)$, donde G es el esquema global que se accede, S es el conjunto de esquemas fuente expresado en un lenguaje y M es la asignación entre G y S, constituida por un conjunto de sentencias. Intuitivamente, el esquema fuente describe la estructura de las fuentes, donde están los datos reales, mientras que el

esquema global proporciona una visión conciliadora, integrada y virtual de las fuentes subyacentes. Las declaraciones en la asignación establecen la conexión entre los elementos del esquema global y los del esquema fuente. Las consultas sobre el sistema se plantean en términos del esquema global *G* y están destinadas a especificar qué datos extraer de la base de datos virtual constituida por el sistema de integración.

1.2.1 Arquitectura de los sistemas de integración

Muchos de los prototipos recientes para la integración de datos comparten una arquitectura común, como se muestra en la figura 1.1.

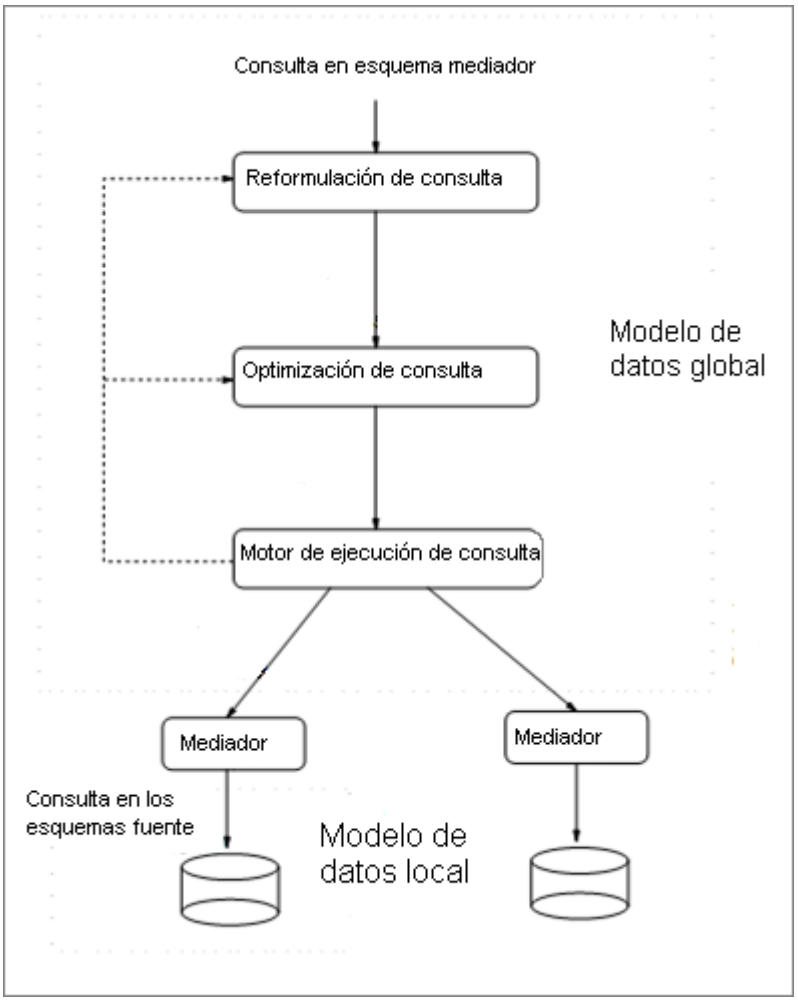

Figura 1.1 Arquitectura prototípica de un sistema de integración de datos

Existen dos características principales que distinguen estos sistemas de un sistema de base de datos tradicional:

- Un sistema de integración de datos se comunica con una fuente de datos usando un conjunto de *mediadores.* Un mediador es un programa que es especifico para cada fuente de datos, y su tarea es la de traducir los datos

en las fuentes de datos a una forma que pueda ser posteriormente procesada por el procesador de consulta del sistema.

- La segunda diferencia con los sistemas tradicionales es que el usuario no hace las consultas directamente al esquema donde se almacenan los datos. La razón es que uno de los objetivos principales de un sistema de integración de datos es liberar al usuario de tener que conocer sobre las fuentes de datos específicas e interactuar con cada una. En su lugar, el usuario plantea las consultas al *esquema mediador.* Este esquema es un conjunto de relaciones virtuales, en el sentido de que estas no están almacenadas en ningún lugar.

Como consecuencia de la distinción entre los esquemas mediadores y los esquemas fuente, el sistema de integración de datos debe primero reformular una consulta del usuario en una consulta que se refiera directamente a los esquemas en las fuentes.

1.3 El proceso de integración

Según (Batini et al., 1986) existen 12 metodologías completas diferentes para llevar a cabo la tarea de la integración. Cada metodología sigue su propio procedimiento de solución. Sin embargo cualquier metodología puede ser considerada eventualmente como una combinación de las actividades siguientes:

Pre integración

Se realiza un análisis de los esquemas antes de la integración para decidir sobre alguna política de integración. Este proceso se encarga de la elección de los esquemas a ser integrados, el orden de la integración, y una posible asignación de preferencias a los esquemas completos o a porciones de esquemas.

Las estrategias globales para la integración, como el número de esquemas a ser integrados a la vez, también se decide en esta fase.

Comparación de esquemas

Los esquemas son analizados y comparados para determinar las correspondencias entre conceptos y detectar posibles conflictos. Las propiedades inter-esquema pueden ser descubiertas en la comparación de esquemas. Se distinguen dos tipos de conflictos: conflictos de nombre y conflictos estructurales.

Conformación de los esquemas

Una vez que los conflictos se detectan se intenta resolverlos de modo que sea posible la combinación de diferentes esquemas. La resolución de conflicto automática generalmente no es factible; se requiere la interacción estrecha con los diseñadores y usuarios antes de que se cumplan los compromisos en cualquier actividad de integración de la vida real.

El objetivo de esta actividad es conformar o alinear los esquemas para hacerlos compatibles para la integración.

Combinación y reestructuración

Con la superposición de esquemas surgen algunos esquemas intermedios integrados. Los resultados intermedios se analizan y, si es necesario se reestructuran a fin de lograr varias características deseadas.

1.3.1 Técnicas para la correspondencia de esquemas

En (Shvaiko and Euzenat, 2005) se propone una clasificación de las técnicas para la correspondencia de esquemas, introduciendo un nuevo criterio basándose en (i) las propiedades generales de las técnicas de asignación, i.e. distinguen entre técnicas aproximadas y exactas; (ii) la interpretación de la información de entrada, i.e. distinguen entre las técnicas sintácticas, externas, semánticas por elemento y de estructura por nivel; y (iii) en el tipo de información de entrada, i.e. distinguen entre técnicas terminológicas, estructurales y semánticas.

1.3.1.1 Técnicas de nivel de elemento

Las técnicas basadas en cadena se usan generalmente para hacer corresponder nombre y descripciones de nombre de entidades esquema/ontologías. Estas técnicas consideran las cadenas como una secuencia de letras en un alfabeto. Están basadas típicamente en la intuición siguiente: mientras más similares sean las cadenas, más probable será que representen los mismos conceptos. Algunos ejemplos de técnicas basadas en cadena que se usan ampliamente en los sistemas de correspondencia son *preffix, suffix, edit distance* y *n-gram*.

Las técnicas basadas en el lenguaje consideran los nombres como palabras en algún lenguaje natural (por ejemplo Inglés). Están basadas en técnicas de Procesamiento del Lenguaje Natural (NLP, por sus siglas en inglés) utilizando las propiedades morfológicas de las palabras de entrada.

Las técnicas basadas en restricciones son algoritmos que tratan con las restricciones internas aplicadas a las definiciones de las entidades, tales como tipos, cardinalidad de los atributos y llaves.

Recursos lingüísticos tales como los tesauros de conocimiento general o de dominio específico que se usan con el fin de hacer corresponder palabras (en este caso los nombres de entidades esquema/ontología se consideran palabras de un lenguaje natural) basándose en relaciones lingüísticas entre ellas (por ejemplo sinónimos, hipónimos).

Las técnicas de reúso de la alineación representan una vía alternativa para el aprovechamiento de los recursos externos, los cuales contienen en este caso alineaciones de esquemas/ontologías acoplados previamente. El reúso de la alineación está motivado por la intuición de que muchos esquemas/ontologías que se requiere hacer corresponder son similares a esquemas/ontologías ya acoplados especialmente si ellos describen el mismo dominio de aplicación. Estas técnicas son particularmente prometedoras cuando se trata con amplios esquemas/ontologías compuestos por cientos y miles de entidades.

1.3.1.2 Técnicas de estructura por nivel

Las técnicas basadas en grafos son algoritmos de grafos que consideran la entrada como grafos etiquetados. Las aplicaciones (por ejemplo esquemas de bases de datos, taxonomías u ontologías) son vistas como estructuras similares a grafos que contienen términos y sus interrelaciones. Usualmente, la comparación de similitud entre un par de nodos de los dos esquemas/ontologías se basa en el análisis de su posición dentro del grafo. La intuición detrás de esto es que, si dos nodos de dos esquemas/ontologías son similares, los nodos vecinos pueden ser también de algún modo similares.

Las técnicas basadas en taxonomías son también algoritmos de grafo que consideran solamente la relación de especialización. La intuición detrás de esta técnica es que los enlaces *is-a* conecta términos que son de por sí similares (siendo uno un subconjunto o un súper conjunto del otro), por tanto sus vecinos ser también de alguna forma similares.

Los repositorios de estructuras almacenan juntos esquemas/ontologías y sus fragmentos con similitudes por pares entre ellos.

Los algoritmos basados en el modelo manejan la entrada basándose en su interpretación semántica. Por tanto, son métodos deductivos bien fundamentados.

1.3.2 Enfoques para la asignación en la integración de datos

Uno de los aspectos más importantes en el diseño de un sistema de integración de datos es la especificación de la correspondencia entre los datos de las fuentes y los del esquema global. Esta correspondencia es modelada mediante la noción de mapeo. Es exactamente esta correspondencia la que determinará como son respondidas las consultas planteadas al sistema mediante el procesador de consultas. Existen dos enfoques importantes en las metodologías de integración de datos para definir este mapeo: global as view (GAV) y local as view (LAV). En (McBrien et al., 2003) se propone un nuevo enfoque llamado both as view (BAV).

1.3.2.1 Global as view

En el enfoque GAV, para cada relación R en el esquema mediador, se escribe una consulta sobre las relaciones fuentes especificando como obtener las tuplas *R's* desde las fuentes.

Por ejemplo, suponga que se tienen dos fuentes de datos BD_1 y BD_2 conteniendo título, escritor, y género de libros. Se puede describir la relación entre las fuentes y la relación ESCRITORLIBRO del esquema mediador como sigue:

BD_1 (id, título, escritor, género) => ESCRITORLIBRO (título, escritor)

BD_2 (id, título, escritor, género) => ESCRITORLIBRO (título, escritor)

Si se tuviera una tercera fuente que comparte los identificadores de libros con BD_1 y provee reseñas de libros, la sentencia siguiente describe como obtener tuplas de la relación RESEÑALIBRO:

BD_1 (id, título, director, género) $\wedge$ BD_3 (id, reseña) =>
RESEÑALIBRO (título, reseña)

En general, las descripciones GAV son reglas que tienen una relación en el esquema mediador en el consecuente, y una conjunción de átomos sobre las relaciones fuentes en el antecedente.

Luego de que se le haya planteado la consulta al esquema mediador, la reformulación de consulta en GAV es relativamente simple. Puesto que las relaciones en el esquema mediador están definidas en términos de las relaciones

fuentes, solo se necesita desplegar las definiciones de las relaciones del esquema mediador. Por ejemplo, suponga que se tiene una consulta para hallar reseñas de libros escritos por John Steinbeck:

q (título, reseña):– ESCRITORLIBRO (título, 'Steinbeck'),
RESEÑALIBRO (título, reseña).

Desplegando las descripciones de ESCRITORLIBRO y RESEÑALIBRO genera las consultas siguientes sobre las relaciones fuente.

q (título, reseña):– DB_1 (id, título, 'Steinbeck', género), DB_3 (id, reseña)
q (título, reseña):– DB_1 (id, título, 'Steinbeck', género),
DB_2 (id, 'Steinbeck', género), DB_3 (id, reseña)

1.3.2.2 Local as view

En el enfoque LAV las descripciones de las fuentes son dadas en la dirección opuesta. Esto es, el contenido de las fuentes de datos es descrito como una consulta sobre la relación del esquema mediador.

Suponga que se tienen dos fuentes (1) V_1, conteniendo títulos, escritores, años de novelas policíacas americanas escritas después de 1980, y (2) V_2, conteniendo reseñas de libros escritos después de 1990. En LAV, se describirían estas fuentes mediante las fórmulas siguientes (las variables que aparecen solo al lado derecho son asumidas como cuantificadas existencialmente):

F_1: V_1 (título, año, escritor) => LIBRO (título, año, escritor, género) Λ
AMERICANO (escritor) Λ
año ≥ 1980 Λ genero= Policíaca.
F_2: S_2 (título, reseña) => LIBRO (título, año, escritor, género) Λ año ≥ 1990 Λ
RESEÑALIBRO (título, reseña).

La reformulación de consulta en LAV es más complicada que en GAV, porque no es posible simplemente desplegar las definiciones de las relaciones en el esquema mediador. Por ejemplo, suponga que nuestra consulta pide reseñas de novelas policíacas escritas después de 1970:

q (título, reseña):– LIBRO (título, año, escritor, Policíaca), año ≥ 1970,
RESEÑALIBRO (título, reseña).

La consulta reformulada sobre las fuentes seria:

q' (título, reseña) :– V_1 (título, año, escritor), V_2 (título, reseña).

Note que, en este caso, la consulta reformulada no es equivalente a la consulta original, porque solo retorna libros que fueron escritos después de 1990.

1.3.2.3 Both as view

En el sistema de integración de datos heterogéneos Automed (*Automatic Generation of Mediator Tools for Heterogeneous database Integration*) (McBrien et al., 2003) se presenta un nuevo enfoque para la integración de datos que engloba los enfoques previos *local as view* (LAV) y *global as view* (GAV). A este método se le denominó con el término *both as view* (BAV), está basado en el uso de secuencias de transformación de esquema reversibles, llamadas sendas de transformación.

En el enfoque BAV la integración de esquemas se especifica como una secuencia de pasos de transformaciones bidireccionales, incrementalmente adicionando, eliminando o renombrando construcciones de manera que se haga corresponder un esquema con otro. Hay asociada opcionalmente con cada paso de transformación una expresión de consulta, especificando la extensión de la construcción nueva o la eliminada en términos del resto de las construcciones del esquema, es decir, describen cómo las instancias de la construcción pueden ser obtenidas a partir de otras construcciones en el esquema, y esta será utilizada durante el procesamiento de consulta. La ausencia de una consulta indica que no se pueden derivar instancias de la construcción a partir de las otras construcciones en el esquema. Cada transformación primitiva tiene una transformación inversa automáticamente derivable. Esta reversibilidad de transformaciones de esquema permite la traducción automática de consulta entre los esquemas.

1.3.2.4 Comparación entre LAV, GAV y BAV

La principal ventaja de GAV sobre LAV es que la reformulación de consulta es muy simple, porque este la reduce al despliegue de las reglas. Sin embargo, agregar

fuentes al sistema de integración de datos no es trivial. En particular, dado una nueva fuente, se necesita resolver todas las vías en las cuales esta puede ser usada para obtener tuplas para cada una de las relaciones en el esquema mediador. Por tanto se necesita considerar la posible interacción de la nueva fuente con cada una de las fuentes existentes, y esto limita la capacidad del enfoque GAV para ajustarse a una colección amplia de fuentes.

En contraste, en el enfoque LAV cada fuente se describe de forma aislada. Es la tarea del sistema (en tiempo de consulta) para resolver cómo las fuentes interactúan y cómo pueden ser combinados sus datos para responder la consulta. La desventaja es que la reformulación de consulta es más difícil, y algunas veces requiere consultas recursivas sobre las fuentes. Una ventaja del enfoque LAV es que es más fácil especificar abundantes restricciones en los contenidos de una fuente (simplemente especificando más condiciones en las descripciones de las fuentes).

BAV es un lenguaje de integración de datos más expresivo que LAV o GAV, ya que toma en cuenta la expresión de las correspondencias en ambas direcciones, y además no está limitado en cuanto a la cantidad de esquemas fuente que están asociados por una correspondencia.

Una ventaja mayor de BAV sobre GAV y LAV es que este soporta fácilmente la evolución tanto de los esquemas fuente como del global, permitiendo que las sendas y los esquemas sean modificados incrementalmente, en oposición a tener que ser regenerados.

1.4 Los dominios de aplicación

Actualmente, la industria de la integración de datos es conocida como Empresa de la Integración de Información (*Enterprise Information Integration*, EII). La visión detrás de esta industria es proporcionar herramientas para la integración de datos a partir de múltiples fuentes sin tener que primero cargar todos los datos en un almacén central como se había requerido en soluciones previas.

Varios factores han llegado a la vez para contribuir al desarrollo de la industria EII, como que las demandas de la gestión de datos en las organizaciones ha cambiado ante la necesidad de crear sitios web externos coherentes requiriendo la integración de datos de múltiples fuentes, y el surgimiento del XML que ha despertado el apetito de las personas de compartir información.

Las arquitecturas subyacentes de los productos han estado basadas en principios similares. Un escenario de integración de datos comienza con la identificación de las fuentes de datos que participaran en la aplicación. Dado que los esquemas son desarrollados de forma independiente, a menudo tienen diferente estructura y terminología. Obviamente, esto puede producirse cuando los esquemas son de distintos ámbitos, tales como un esquema de bienes raíces y el esquema de impuestos a la propiedad. Sin embargo, también se produce incluso si se modela el mismo dominio del mundo real, sólo porque han sido desarrollados por diferentes personas en diferentes contextos del mundo real. De modo que el primer paso en la integración de esquemas es identificar y caracterizar estas relaciones inter-esquemáticas. Este es el proceso de correspondencia de esquemas. Luego se construye un esquema mediador (generalmente llamado *esquema virtual*) el cual será consultado por usuarios o aplicaciones, y se crean asignaciones semánticas desde las fuentes de datos al esquema mediador. El procesamiento de consulta debe comenzar por la reformulación de una consulta planteada al esquema virtual en consultas sobre las fuentes de datos, y luego ejecutándola eficientemente con un generador que crea estrategias que cubren múltiples fuentes de datos y tratan con las limitaciones y capacidades de cada fuente.

Algunas de las compañías coincidieron con el surgimiento del XML, y construyeron sus sistemas en un modelo de datos y lenguaje de consulta XML. Estas compañías tuvieron que enfrentar un conjunto de problemas adicional en comparación con otras compañías, ya que la investigación del procesamiento eficiente de consultas y la integración para XML estaba solo en sus primeros pasos y por tanto no tenían una literatura amplia en la que basarse.

Como sucede con cualquier industria nueva, EII ha confrontado muchos retos, algunos de los cuales aún hoy impiden su crecimiento.

1.4.1 Almacenes de datos

Una variante del problema de integración de esquemas que se hizo muy popular en la década de 1990 es el de integrar fuentes de datos en un almacén de datos. Un almacén de datos es una colección de datos orientada a un determinado ámbito (empresa, organización, etc.), integrado, no volátil y variable en el tiempo, que ayuda a la toma de decisiones en la entidad en la que se utiliza. El proceso de extracción requiere la transformación de los datos del formato de la fuente en el

formato del almacén. Como se muestra en (Bernstein and Raham, 2000), la operación de correspondencia es útil para el diseño de las transformaciones. Dada una fuente de datos, un enfoque para la creación de transformaciones adecuadas es empezar por encontrar aquellos elementos de la fuente que también están presentes en el almacén. Se trata de una operación de correspondencia.

Después que un mapeo inicial se crea, el diseñador del almacén de datos tiene que examinar la semántica detallada de cada elemento fuente y crear transformaciones que concilien esas semánticas con aquellas del destino.

1.4.2 Comercio electrónico

En la presente década, el comercio electrónico ha dado lugar a una nueva motivación para la correspondencia de esquemas: la traducción de los mensajes. Con frecuencia los socios comerciales intercambian mensajes que describen las transacciones comerciales. Por lo general, cada socio comercial utiliza su propio formato de mensaje. Los formatos de mensaje pueden variar en su sintaxis, como las estructuras EDI (intercambio electrónico de datos), XML. También pueden usar diferentes esquemas de mensajes.

Para permitir que los sistemas puedan intercambiar mensajes, los desarrolladores de aplicaciones necesitan convertir mensajes entre los formatos requeridos por los diferentes socios comerciales. Parte del problema de traducción de mensajes es la traducción entre los diferentes esquemas de mensajes. Los esquemas de mensaje pueden utilizar nombres diferentes, tipos de datos un tanto diferentes, y diferentes rangos de valores permitidos. Los campos se agrupan en estructuras que también pueden diferir entre los dos formatos. Por ejemplo, uno puede ser una estructura plana que se limita a enumerar los campos, mientras que otro puede agrupar campos relacionados. O ambos formatos pueden utilizar estructuras anidadas, pero pueden agrupar los campos en diferentes combinaciones.

La traducción entre diferentes esquemas de mensaje es, en parte, un problema de correspondencia de esquemas. Hoy en día, los diseñadores de aplicaciones necesitan especificar de forma manual como están relacionados los formatos de mensajes. Una operación de correspondencia reduciría la cantidad de trabajo manual generando un diseño de mapeo entre los dos esquemas de mensajes, que un diseñador de aplicaciones posteriormente puede validar y modificar según sea necesario.

La correspondencia de esquemas también puede ser útil para aplicaciones que están consideradas para la web semántica (Berners-Lee et al., 2001), tales como el mapeo de mensajes entre agentes autónomos o la correspondencia de definiciones declarativas mediadoras.

1.4.3 Bases de datos P2P

Las redes P2P (*Peer to Peer*) se caracterizan por una flexibilidad y una dinámica extremas. Los clientes pueden aparecer y desaparecer en la red, sus bases de datos son autónomas en su lenguaje, contenido, y pueden cambiar sus esquemas, etcétera. Debido a que los clientes son autónomos, podría usar una terminología diferente, incluso si se refieren al mismo dominio de interés. Por tanto, con el fin de establecer el intercambio de información (sustanciosa) entre los clientes, uno de los pasos es identificar y caracterizar las relaciones entre sus esquemas. Una vez identificadas estas relaciones, el próximo paso es usarlas para el propósito de respuesta a consulta, por ejemplo, utilizando técnicas de la integración de datos como LAV o GAV. Sin embargo, las aplicaciones P2P plantean requerimientos adicionales en los algoritmos de correspondencia. En un ambiente P2P la suposición de que todos los clientes cuentan con un esquema global no puede hacerse, ya que el este esquema global necesitaría actualizarse cada vez que el sistema se desarrolle. De este modo, si en el caso de la integración de datos la operación de correspondencia de esquemas puede ser realizada en tiempo de diseño, en las aplicaciones P2P los clientes necesitan coordinar sus bases de datos sobre la marcha, requiriendo por tanto una operación de correspondencia de esquemas en tiempo de ejecución.

1.5 Trabajos relacionados

El problema de la integración de datos ha sido tratado en dos contextos principales: la integración tradicional de esquemas en bases de datos heterogéneas y la integración de datos semiestructurados. En ambos contextos se requiere primero la identificación de las relaciones semánticas entre las fuentes de datos, aspecto que ha sido ampliamente estudiado. Se han hecho varias contribuciones en cuanto a la representación de las fuentes de datos y las técnicas de asignación entre el esquema global y las fuentes de datos. A continuación se ofrecen las propuestas de algunas de las investigaciones, proyectos y sistemas más relevantes en esta área.

1.5.1 Integración de información heterogénea

El análisis, descubrimiento, y representación de las propiedades inter esquemas es un aspecto crítico del proceso de integración y en la literatura se han propuesto muchos enfoques para ejecutar la extracción de las propiedades inter esquemas.

En Cupido (Madhavan et al., 2001), se presenta un sistema para la derivación de propiedades inter esquemas entre fuentes de información heterogéneas. La derivación de las propiedades se lleva a cabo mediante la realización de dos tipos de exámenes, llamadas correspondencias lingüísticas y de estructura. Los esquemas de entrada se codifican como grafos. Los nodos representan los elementos del esquema y se cruzan en una manera combinada hacia arriba y hacia abajo. El algoritmo de correspondencia consiste en tres fases y opera solo con estructuras de árboles.

En (Palopoli et al., 1998), se describen técnicas semiautomáticas para descubrir sinónimos, homónimos y relaciones de inclusión de objeto en los esquemas de base de datos. Es importante notar que el diseño de sistemas para la recopilación de información de múltiples fuentes también ha sido tema de investigación en Inteligencia Artificial a través de los sistemas multiagentes (Lesser et al., 1998), concentrándose principalmente en las tareas de alto nivel relacionadas con el proceso de extracción (Dragoni, 1997).

Se han desarrollado también muchos proyectos de integración de datos basados en una arquitectura mediadora. Por ejemplo el proyecto TSIMMIS (Chawathe et al., 1994) sigue un enfoque estructural y usa un modelo de auto descripción (OEM) para representar fuentes de datos heterogéneas y técnicas de correspondencia de patrones para ejecutar un conjunto de consultas predefinido basado en una plantilla de consultas.

El proyecto *Garlic* (Carey et al., 1994) se construye sobre una arquitectura de envoltura compleja para describir las fuentes locales con un lenguaje OO (GDL), y sobre la definición de Objetos Complejos *Garlic* para unificar manualmente las fuentes locales y entonces definir un esquema global.

El proyecto SIMS (Arens et al., 1996) propone crear una definición de esquema global haciendo uso de Lógica Descriptiva (i.e. el lenguaje LOOM) para describir las fuentes de información.

En el sistema OBSERVER se toma un enfoque basado en Lógica Descriptiva y ontologías para dar soporte a la interoperación semántica y a la formulación de consultas sobre repositorios de información distribuida donde se usan diferentes vocabularios (Mena et al., 1996). Aquí la idea es que cada repositorio tiene su propia ontología. Las relaciones inter-ontologías se especifican en una forma declarativa (usando Lógica Descriptiva) en un módulo gestor para manipular heterogeneidades del vocabulario entre ontologías de diferentes repositorios de información para el procesamiento de consultas.

1.5.2 Integración de datos semiestructurados

La cuestión del modelado e integración de los datos semiestructurados ha sido investigada en la literatura reciente. En particular, un sondeo de los problemas que tratan el modelado y consulta de datos semiestructurados se presenta en (Buneman, 1997).

Clio (Popa et al., 2002) traduce los esquemas de fuentes de datos, XML o relacionales, en una representación interna. Luego de que las asignaciones entre los esquemas fuente y objetivo se hayan derivado semiautomáticamente, el esquema objetivo se materializa a partir de los datos de las fuentes, utilizando las asignaciones y un conjunto de reglas.

Xyleme (Reynaud et al., 2001) también adopta un enfoque basado en asignaciones. Este usa un árbol como esquema global y los esquemas fuentes son asignados a este árbol a través de caminos de asignación.

DIXSE (Rodriguez-Gianolli and Mylopoulos, 2001) transforma las especificaciones DTD de un conjunto de documentos fuente XML en una representación conceptual interna, usando una heurística para capturar la semántica.

1.5.2.1 Automed

Automed es un sistema de integración de datos heterogéneos que ha desarrollado la primera implementación (Boyd et al., 2004) de la técnica de integración de datos BAV (McBrien, 2003), la cual abarca el poder expresivo de otras técnicas de integración de datos publicadas GAV y LAV. Se distingue por tener una metodología clara para manipular un amplio rango de lenguajes de modelado de datos en el proceso de integración, a diferencia de otros enfoques que asumen que la integración siempre es realizada en un único modelo de datos común. Esto se logra permitiendo relacionar las construcciones de modelado de un lenguaje de

modelado como Entidad Relación (ER), UML, relacional o XML, con las construcciones en un único lenguaje de modelado de datos común de bajo nivel denominado el modelo de datos de hipergrafo (HDM). Un esquema HDM consta de un conjunto de nodos, aristas y restricciones, y cada construcción de modelado de un lenguaje de modelado de alto nivel

La metodología de integración de datos XML en Automed es similar al proceso de cuatro pasos descrito en (Batini et al., 1986). Inicialmente, los esquemas se comparan para identificar las relaciones semánticas (Rizopoulos, 2003) entre sus elementos, lo que se denomina *correspondencia de esquemas,* utilizando por ejemplo, técnicas de minería de datos o correspondencias semánticas con ontologías. Luego sigue la fase de *conformación de esquemas* donde se resuelven los conflictos de nombre mediante transformaciones de renombre BAV. Entonces los esquemas se superponen por separado transformándose y produciendo dos esquemas unión-compatible, los cuales se definen para que sean idénticos mediante una secuencia de transformaciones BAV denominadas *id*. Id es un tipo especial de transformación primitiva que hace corresponder dos construcciones sintácticamente idénticas en dos esquemas unión diferentes, significando su equivalencia semántica. Este proceso es llevado a cabo por un algoritmo que puede ser aplicado de dos formas: hacia arriba o hacia abajo, y que crea automáticamente la vía de transformación entre un esquema de fuente de datos y su esquema-unión correspondiente. Este algoritmo se basa en la restructuración de Esquemas de Fuentes de Datos XML. Uno de los esquemas-unión es seleccionado arbitrariamente para que sea el esquema intermedio, el cual en la fase final de *combinación de esquemas* se transforma basado en las relaciones de inclusión, solapamiento y exclusión para producir el esquema integrado final.

1.6 Conclusiones Parciales

Hasta hace poco, la integración de datos era considerada un área para la curiosidad intelectual, pero actualmente es una necesidad. La economía actual, basada en una vasta infraestructura de redes de computadoras y la capacidad de las aplicaciones de intercambiar datos en XML, acentúa aun más la necesidad de soluciones de integración de datos.

En el proceso de integración se destacan aspectos como la correspondencia de esquemas y la asignación entre el esquema global y las fuentes, para los cuales se

han propuesto diferentes técnicas y enfoques. Para tratar el problema de la asignación se han planteado dos enfoques principales: LAV y GAV, pero ambos tienen inconvenientes que limitan su capacidad para brindar buenas soluciones. Un tercer enfoque BAV que ha sido propuesto por el proyecto Automed fue concebido para cubrir las limitaciones de LAV y GAV.

Se han desarrollado varios sistemas y proyectos que adoptan metodologías diferentes para la integración de datos, sin embargo en el área de la integración de datos semiestructurados, como XML, aún existen dificultades que impiden su implementación.

El proyecto Automed ha desarrollado la primera implementación de la técnica de integración BAV, permitiendo la manipulación de un amplio rango de lenguajes de modelado de datos incluido XML. El proceso de integración comienza con la correspondencia de esquemas, luego se realiza la conformación de esquemas y finalmente la combinación y restructuración donde se aplica un algoritmo de restructuración, obteniéndose el esquema global.

CAPÍTULO II. **AUTOMED**

En este capítulo se describen la arquitectura y etapas del proceso de integración del sistema Automed. Además se presentan las contribuciones y funcionalidades que ofrece para la integración de datos XML, incluyendo las Interfaces de Programación de Aplicación (API) que se utilizan para desarrollar una herramienta que dé solución al problema de la integración de esquemas XML heterogéneos.

2.1 Cuestiones generales sobre Automed

El proyecto Automed ha desarrollado un conjunto de herramientas para dar soporte al enfoque de integración de datos *both as vew* (BAV).

Un sistema de integración de datos proveerá una vista unificada de un número de fuentes de datos, haciéndolas parecer como una fuente de datos única para un usuario o un programa de aplicación. Este proceso de integración de datos tiene dos aspectos principales: integración de esquemas, donde las estructuras de los diferentes esquemas locales de fuentes de datos son lógicamente relacionados con un único esquema global por medio de un conjunto de correspondencias, y procesamiento de consultas donde las consultas y las actualizaciones en un esquema se hacen corresponder con consultas y actualizaciones sobre los otros esquemas.

2.1.1 Escenario de integración general

En Automed cada fuente de datos se describe por un esquema, el cual es transformado en un esquema unión-compatible por medio de una serie de transformaciones primitivas reversibles, creando en consecuencia una senda de transformaciones entre una fuente de datos y su respectivo esquema unión-compatible. Todos los esquemas unión son sintácticamente idénticos y esto se especifica por una serie de transformaciones id entre cada par de esquemas unión. Id es un tipo especial de transformaciones primitivas que hace corresponder dos construcciones sintácticamente idénticas en dos esquemas unión diferentes, indicando su equivalencia semántica. La senda de transformación conteniendo estas transformaciones id puede ser generada automáticamente. Uno de los esquemas unión arbitrario puede entonces designarse como el esquema global, o seleccionarse para transformaciones adicionales en un nuevo esquema que se convertiría en el esquema global.

La transformación de una fuente de datos en un esquema unión se logra aplicando una serie de transformaciones primitivas, haciendo cada una un cambio “delta” al esquema actual agregando, eliminando o renombrando una construcción de esquema. Cada transformación *adición* y *eliminación* está acompañada por una consulta especificando la extensión de la nueva construcción agregada o eliminada, es decir como los datos asociados a esta construcción pueden derivarse de otras construcciones en el esquema original. Esta consulta se expresa en el Lenguaje de Consulta Intermedio de Automed, IQL [23,12]. El resultado es una secuencia de esquemas intermedios, conectando cada esquema de datos fuente con su respectivo esquema unión. La consulta dada con una transformación primitiva define la nueva o eliminada construcción de esquema en términos de las otras construcciones de esquema, y por tanto brinda la información necesaria para hacer las transformaciones primitivas automáticamente reversibles. Esto identifica a Automed como un sistema de integración de datos BAV, que incluye los enfoques LAV y GAV, tanto como sea posible para extraer una definición del esquema global como una vista sobre los esquemas de datos fuente, y es también posible extraer definiciones de los esquemas de datos fuente como vistas sobre el esquema global.

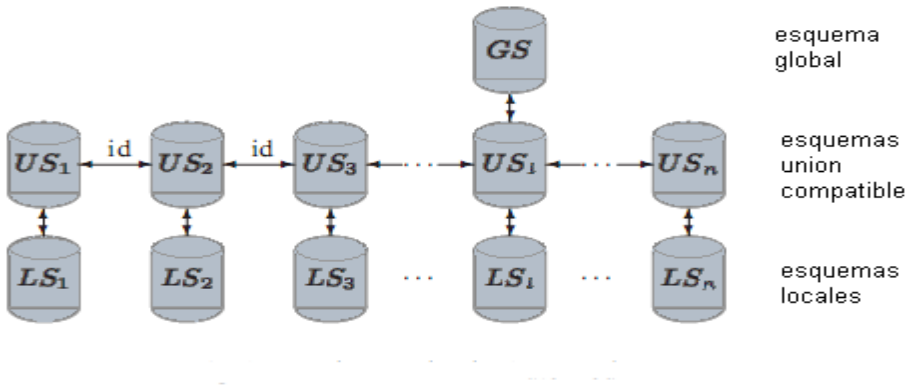

Figura 2.1 Integración de datos en Automed

En la figura 2.1 cada esquema unión puede contener información que no puede derivarse del esquema local correspondiente. Estas construcciones no son insertadas en los esquemas unión a través de una transformación de *adición*, sino más bien mediante una transformación *extend* de la nueva construcción. Las transformaciones *extend y contract* tienen un comportamiento similar a *adición* y *eliminación* respectivamente excepto que ellas indican que su consulta

acompañante construye solo parcialmente la extensión de la construcción de esquema nueva o eliminada. Cada consulta de la transformación *extend/contract* tiene la forma *Rango* q_L q_U, especificando un límite inferior y uno superior. El límite inferior puede ser Void y el límite superior puede ser Any, los cuales indican respectivamente información no conocida sobre el límite inferior o superior de la extensión de la nueva construcción. Pude haber también información presente en el esquema de fuentes de datos local que podría no estar presente dentro del esquema unión correspondiente, y esto es eliminado con una transformación *contract*, en vez de una transformación *eliminación*.

2.2 Arquitectura de Automed

En el enfoque BAV, la integración de esquemas es especificada como una secuencia de pasos de transformación bidireccionales, incrementalmente adicionando, eliminando o renombrando construcciones, de modo que se asigne un esquema a otro. Opcionalmente asociada a cada paso de transformación hay una expresión de consulta, describiendo cómo las instancias de la construcción pueden ser obtenidas a partir de las otras construcciones en el esquema, la cual será usada durante el procesamiento de consulta. La ausencia de una consulta indica que no hay instancias de la construcción que puedan derivarse de las otras construcciones en el esquema. Una de las nuevas características del enfoque usado en Automed es que no está restringido a utilizar un modelo de datos común (CDM) particular para la integración de datos. En su lugar, funciona bajo el principio de que los lenguajes de modelado de datos como ER, relacional, UML, XML, etc. son modelos de datos basados en grafos, los cuales pueden ser descritos en términos de construcciones en el Modelo de Datos de Hipergrafo (HDM) (Poulovassilis and McBrien, 1998). La implementación de Automed brinda solo soporte directo para el HDM, y es asunto de la configuración de Automed dar soporte a una variante particular de un lenguaje de modelado de datos. Una vez configurado para usar un lenguaje de modelado de datos, los esquemas y las transformaciones sobre estos pueden ser descritos en términos de operaciones sobre las construcciones de ese lenguaje de modelado de datos.

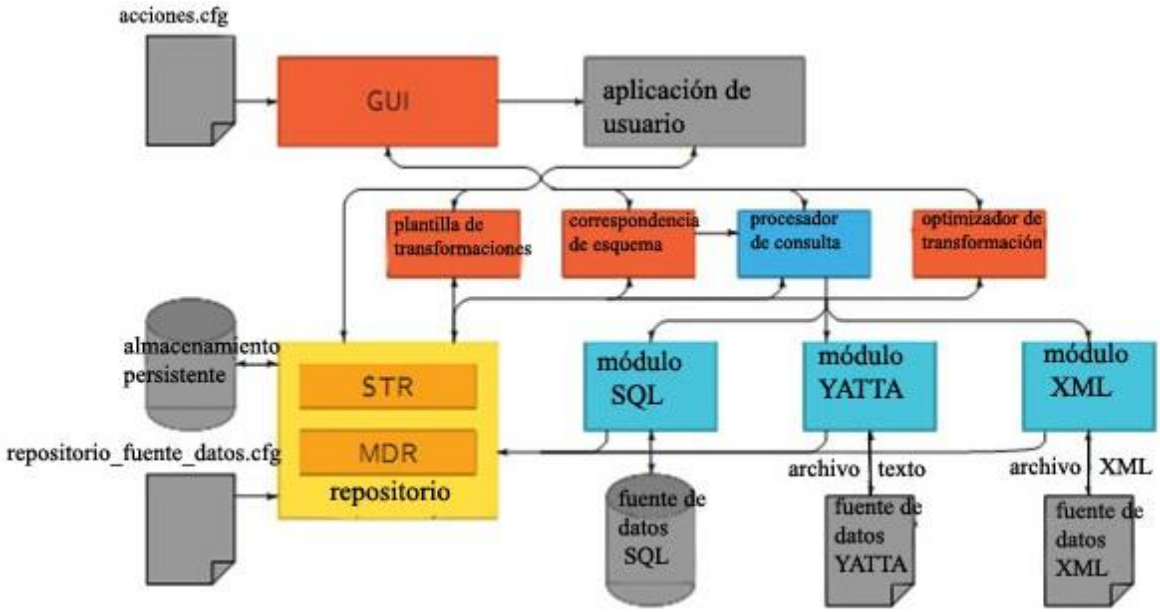

Figura 2.2 Arquitectura de Automed

Cuando una fuente de datos es llevada a representación HDM, una definición del esquema para esa fuente de datos se agrega al repositorio. La herramienta de correspondencia de esquemas puede entonces usarse para identificar objetos relacionados en varias fuentes de datos (accediendo al procesador de consultas para recuperar datos desde los objetos del esquema), y la herramienta de plantilla de transformación usada para generar transformaciones entre las fuentes de datos. Un enfoque típico para tratar con la heterogeneidad de las fuentes de datos selecciona un modelo como un modelo común de datos (Sheth and Larson, 1990) y convierte los lenguajes de modelado de las fuentes de datos en este modelo. En la **Error! Reference source not found.** se muestra una arquitectura de integración. Un mediador (Hammer et al., 1997, Roth and Shwartz, 1997) es un software que traduce entre los lenguajes locales y modelos de fuentes de datos y el lenguaje y modelo común de datos. El mediador transforma subconsultas enviadas a las fuentes de datos y responde las retornadas desde las fuentes de datos. Los mediadores (Wiederhold, 1992) son componentes que obtienen información de las fuentes de datos intermedias u otros mediadores. Estos proveen información a otros mediadores por encima de ellos o a los usuarios del sistema. Los mediadores exportan un esquema mediador el cual es una representación integrada de las fuentes de datos. Los usuarios pueden dirigir sus consultas a los mediadores o a las fuentes intermedias usando el lenguaje del modelo común de datos o usando su propio lenguaje local. Un mediador puede ser interpretado como una vista de los

datos encontrados en uno o más fuentes de datos (Ullman, 1997). Tradicionalmente los datos no se almacenan en un mediador. Los usuarios consultan el esquema mediador, y las vistas transforman estas consultas entre el esquema mediador y las fuentes de datos (Domenig and Dittrich, 1999).

2.2.1 Módulos mediadores Automed

Al existir un amplio rango de lenguajes de modelado de datos en uso, una tarea común en las metodologías de integración de datos es implementar un mediador para traducir todos los esquemas componentes en un modelo común de datos. En Automed cada fuente de datos tiene un mediador para traducirlo a su especificación equivalente en términos del lenguaje común de datos de bajo nivel HDM. Después de que una consulta expresada sobre el esquema integrado sea reformulada y optimizada, sus subconsultas son enviadas a los mediadores de fuentes de datos apropiados para la traducción a los lenguajes de consulta de las fuentes de datos y la evaluación en estas fuentes. Entonces los mediadores traducen de vuelta los resultados de las subconsultas a IQL y el procesador de consulta IQL se encarga de cualquier post procesamiento adicional necesario de la consulta (Poulovassilis, 2001).

2.2.2 Correspondencia de esquemas

Para llevar a cabo el proceso de correspondencia de esquemas es preciso primero identificar las relaciones semánticas entre estos. Se distinguen cinco tipos de relaciones semánticas entre los esquemas (Rizopoulos, 2003): equivalencia, inclusión, intersección, disyunción e incompatibilidad.

Con el fin de descubrir las relaciones semánticas entre los esquemas, en Automed se realiza una comparación bidireccional de sus elementos. La idea detrás de la comparación bidireccional de elementos de esquema viene del hecho de que el número del grado de similitud entre dos elementos esquemas indica su similitud o disimilitud. Suponiéndose que existen dos elementos de esquema A y B, se define como *d(A,B)* el grado de similitud producido por la comparación del elemento A contra el elemento B y *d(B,A)* el grado de similitud producido por la comparación del elemento B contra el elemento A. Se le denomina a *d(A,B)* y *d(B,A) grados de similitud bidireccional*. Intuitivamente, mientras más similar sea A a B, más alto será el grado de similitud *d(A,B)*. El mismo razonamiento se aplica para la comparación inversa de B contra A.

La correspondencia de esquemas en Automed se compone de varios módulos de comparación que utilizan diferentes tipos de información para determinar la similitud de los elementos de esquema. Estos módulos toman como entrada las fuentes de datos y sus instancias de datos y trabajan independientemente para producir grados de similitud bidireccional parciales. Parciales en el sentido de que se producen por la comparación de información parcial, por ejemplo nombre de elementos solamente, y son por tanto parcialmente correctos. Estos grados son combinados luego para proporcionar los grados de similitud finales, que indican las relaciones semánticas entre los elementos de esquema.

Existen dos tipos de módulos: de identificación de relación y de aclaración de relación. Los de identificación intentan descubrir los pares de elementos compatibles y los de aclaración intentan especificar el tipo de relación semántica en cada par compatible.

El Módulo Nombre de Elemento realiza una comparación entre los elementos de esquema que no distingue entre mayúsculas y minúsculas. Cuando A tiene el mismo nombre que B entonces d(A,B)= d(B,A)= 1 y si el nombre de A es una subcadena de B entonces d(A,B)=0.2 y d(B,A)=1. Si un elemento es una subcadena de otro, entonces es un indicio de una relación de inclusión. El Módulo Tipo de Dato compara elementos de tipo de datos. El Módulo de Estadística Numérico compara elementos numéricos. El Módulo de Estadística No-Numérico compara elementos no numéricos basado en el número promedio de apariciones de caracteres especiales (@,$, etc.) en sus instancias. El Módulo Instancias usa un clasificador para identificar similitudes entre elementos comparando sus instancias. El Módulo Número de Instancias es un módulo se usa para la aclaración de la relación y compara el número de instancias de los elementos distintas. El Módulo Precisión es un módulo de aclaración de relación que compara el rango de cada instancia de elemento y el Módulo Longitud compara el rango de las longitudes de los elementos. El Módulo Existencia es un módulo de aclaración de relación que examina la existencia de instancias en elementos.

Una vez identificadas las relaciones semánticas entre los elementos de esquema las sendas de transformación BAV pueden derivarse automáticamente para integrar los esquemas y las fuentes de datos. Este proceso tiene varias etapas: en la fase de conformación de esquemas se resuelven los conflictos de nombre

mediante transformaciones de renombre BAV, los esquemas se superponen por separado transformándose produciendo los esquemas intermedios que entonces son reestructurados mediante el uso de un algoritmo para producir el esquema final.

2.2.3 El procesador de consultas en Automed

El Lenguaje de Consulta Intermedio de Automed (*Automed Intermediate Query Language, IQL* (Poulovassilis, 2001)) es un lenguaje de consulta funcional basado en la comprensión. Su propósito es proporcionar un lenguaje de consulta común de modo que las consultas escritas en lenguajes de consultas de alto nivel diferentes (por ejemplo SQL, XQuery, OQL) puedan ser traducidas hacia y desde este.

El procesamiento de consulta en Automed está estrechamente unido al lenguaje de consulta IQL. Para habilitar el uso de lenguajes de consulta de alto nivel tales como SQL y XQuery con Automed, el componente Traductor de Automed (Automed Translator) puede usarse para traducir una consulta expresada en estos lenguajes de consulta a su consulta IQL equivalente, la que puede entonces ser enviada al Procesador de Consulta de Automed, AQP.

La figura 2.2 muestra los componentes de AQP, nombrados el Reformulador de Consulta, el Optimizador de Consulta, el Anotador de Consulta y el Evaluador de Consulta.

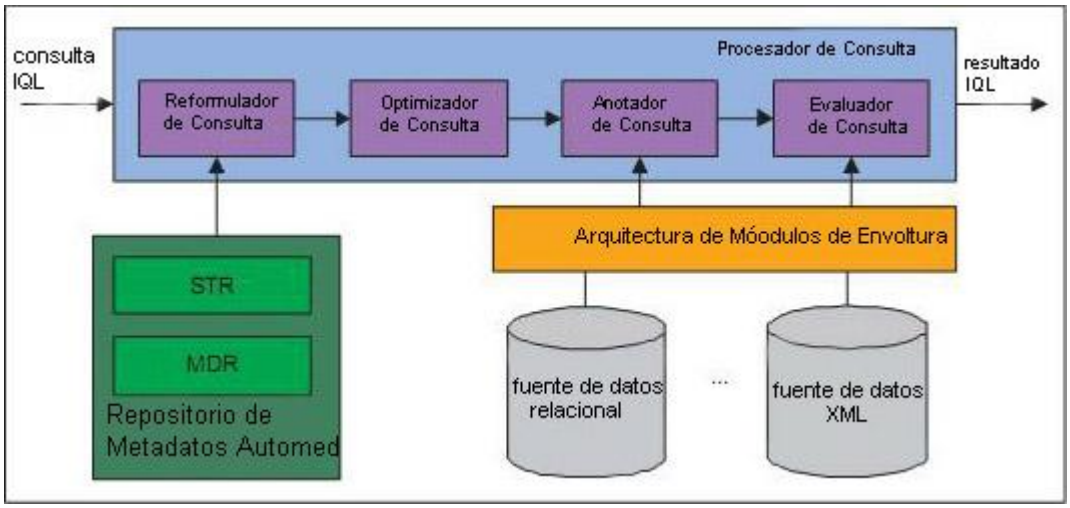

Figura 2.2 El procesador de consulta de Automed.

El AQP es usado para evaluar las consultas presentadas a un esquema virtual contra un conjunto de fuentes de datos. Esto se hace coordinando los componentes de la figura 2.2 pero el AQP es agnóstico de lo que estos contienen.

Cada uno de estos componentes puede tener más de una implementación. La implementación a usar por cada componente se especifica usando el componente de AQP Configuración del Procesador de Consulta. Este configura el AQP así como los componentes que contiene.

La consulta de entrada se reformula primero a una consulta equivalente que solo contiene construcciones de fuente de datos usando el componente Reformulador de Consulta. Entonces el componente Optimizador de Consulta optimiza la consulta reformulada y el componente Anotador de Consulta inserta objetos mediadores de Automed dentro de la consulta optimizada. El evaluador de consulta se usa luego para evaluar la consulta anotada. El resultado de cada una de estas fases del procesamiento de consulta es registrado.

El componente Reformulador de Consulta es capaz de reformular consultas hechas a un esquema virtual contra un conjunto de fuentes de datos extensionales, es decir que existen físicamente, usando reformulación GAV, LAV o BAV.

Este componente está dividido en tres capas con el fin de ser capaz de proporcionar los tres tipos de reformulación. Todas las técnicas de reformulación implementadas producen un mapa de vistas que contiene definiciones de vistas de las construcciones del esquema objetivo en términos de las construcciones del esquema de fuente de datos.

Luego de que la consulta inicial sobre el esquema virtual haya sido reformulada en una consulta conteniendo construcciones de esquema de fuente de datos, el componente de optimización lógica realiza varias optimizaciones lógicas en la consulta. El objetivo de este componente es doble: primero, simplificar la consulta mediante la realización de optimizaciones algebraicas, y, segundo, construir las subconsultas más extensas posibles que puedan ser enviadas a las fuentes de datos locales para la evaluación

Luego de que la consulta de entrada haya sido reformulada y procesada por el optimizador lógico, funciones mediadoras, las cuales son responsables de la evaluación de las consultas IQL en las fuentes de datos, son insertadas dentro de la consulta. Los mediadores son implementados en IQL. Cada tipo de mediador es capaz de traducir un subconjunto del lenguaje IQL.

Evaluar una consulta expresada en un lenguaje funcional consiste en realizar reducciones en expresiones reducibles hasta que no se puedan hacer más

reducciones. Se dice que entonces está en forma normal. Puede que halla más de una expresión reducible dentro de la consulta y por tanto una variedad de reducciones, pero el orden en el cual estas reducciones son ejecutadas no hace diferencia y el resultado es el mismo siempre que la evaluación finalice.

El componente Evaluador siempre reduce la expresión reducible del extremo más a la izquierda –esto es conocido como reducción orden-normal y tiene el mejor comportamiento de finalización posible.

2.2.4 El repositorio Automed para la integración de datos BAV

El repositorio Automed tiene en su núcleo el paquete de Java reps el cual crea una plataforma para que otros componentes de Automed sean implementados sobre él. Todos los esquemas fuente, esquemas intermediarios, esquemas integrados, y las sendas de transformación entre ellos se almacenan en los repositorios de esquema y de transformaciones de Automed.

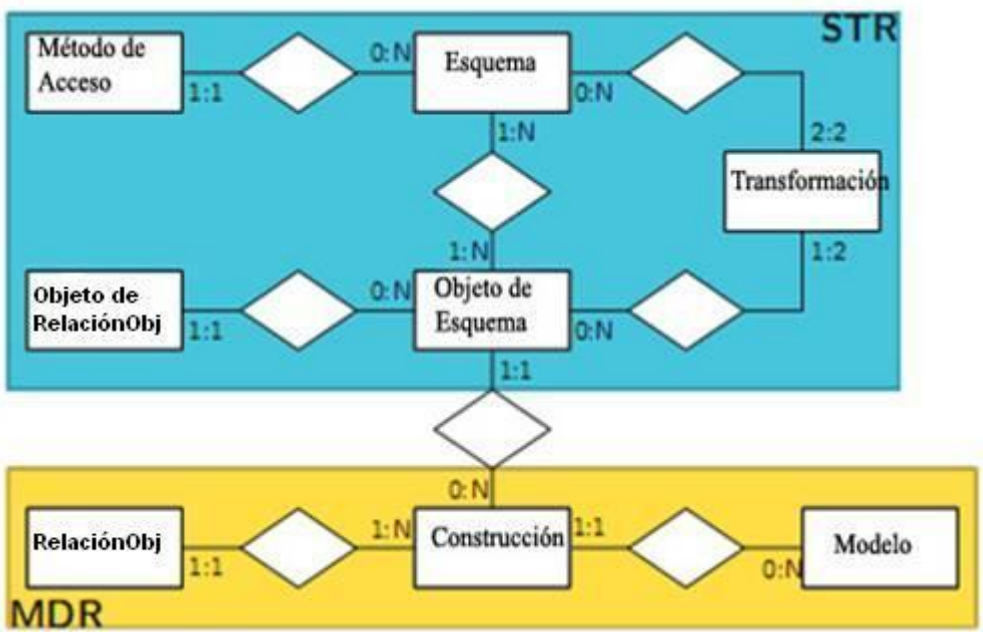

Figura 2.3 Repositorio de Esquemas

El repositorio Automed tiene dos componentes lógicos, a los que se accede a través de las API. El Repositorio de Definición del Modelo (MDR) permite la descripción de cómo se representa un lenguaje de modelado de datos como una combinación de nodos, aristas y restricciones en el HDM. El Repositorio de Transformación de Esquema (STR) permite que los esquemas se definan en términos de los conceptos del modelado de datos en el MDR. Existen dos tipos de esquemas que se almacenan: extensionales e intencionales. Un esquema

extensional es aquel que existe físicamente, mientras que uno intencional es aquel derivado de un esquema extensional u otro intencional, aquí también se especifican las transformaciones entre esos esquemas. En la figura 2.3 se muestra las clases y su relación en el repositorio de esquemas.

2.2.4.1 Información general MDR

El MDR almacena descripciones de lenguajes de modelado. En esencia, está compuesto por una lista de construcciones para cada lenguaje de modelado junto con los argumentos necesarios para crear una instancia de cada construcción y los medios para traducir las instancias de las construcciones en HDM.

Cada argumento necesario para crear una instancia puede ser un simple nombre, una referencia a un objeto existente de un tipo de construcción dado, una lista de alternativas o una secuencia.

La clase Modelo representa los modelos de datos, tales como ER o XML. La clase Construcción representa los tipos de elementos diferentes de un modelo específico, es decir, una entidad en el modelo ER o un elemento en XML. La tabla Objeto almacena detalles de los objetos individualmente, pero no proporciona información sobre la conexión entre los diferentes objetos en el esquema. La tabla RelaciónObj se designa para almacenar esta información.

2.2.4.2 Información general STR

El STR guarda información sobre los esquemas y las sendas de transformación entre ellos. Un esquema es una lista de objetos y relaciones de objetos asociados. La clase Esquema representa los esquemas. A cada esquema entrado en el repositorio se le asigna un Id de esquema (SID) que lo identifica como único en la tabla. Se guarda también información sobre la naturaleza de los esquemas, es decir, si son extensionales y si son una fuente de datos. Cuando se crea un esquema, se crea también un objeto en Objeto de Esquema.

La tabla objeto almacena información sobre cada objeto (o construcción) en un esquema. Los campos Id de modelo e ID de construcción indican el lenguaje de modelado utilizado y el tipo de construcción al que el objeto pertenece en ese lenguaje, respectivamente. De modo que cada objeto se identifica como una instancia de una construcción particular en el MDR.

Las transformaciones entre los esquemas se especifican en la tabla transformación. La especificación de la transformación incluye una acción, que puede ser *adición*, *eliminación*, *renombre*, *extender,* y *contraer*.

La tabla método de acceso guarda información sobre como se hace la conexión con las bases de datos. Cada método de acceso tiene un identificador, un ID de esquema, un identificador de protocolo que indica el protocolo usado para la conexión.

Los esquemas en sí no tienen un lenguaje de modelado inherente. En su lugar, cada objeto, a través de su tipo de construcción, es de un lenguaje de modelado particular. Esto significa que las sendas de transformación que convierten un esquema de un lenguaje a otro pueden ser definidas debido a que los esquemas intermediarios con algunas construcciones de ambos lenguajes pueden ser descritos.

Se almacenan esquemas de fuentes de datos reales (datos extensionales) y esquemas intermediarios (datos intencionales) que existen en las sendas de transformación entre los esquemas de las fuentes de datos reales. Estos esquemas intermediarios son usualmente ellos mismos intencionales; esto es, los esquemas pueden ser descritos solamente mirando la extensión de otro esquema y una senda de transformaciones entre los dos. Los esquemas intencionales pueden ser materializados temporalmente (i.e. hechos extensionales) y de hecho esto ocurre automáticamente cuando la descripción del esquema es solicitada, de manera que estas diferencias son, en su mayoría, transparentes para las herramientas que utilizan los repositorios.

2.3 Escenario de integración de datos XML en Automed

El sistema de integración de datos Automed ha sido extendido para dar soporte a la integración de documentos XML heterogéneos.

Cada fuente de datos XML es descrita por un Esquema de Fuente de Datos XML (que será presentado en la sección 2.3.1), que es transformado en un esquema intermedio, mediante unas series de transformaciones que insertan, eliminan o renombran construcciones de esquema. Los esquemas unión son entonces producidos automáticamente, y ellos extienden cada esquema intermedio con las construcciones del resto de los esquemas intermedios. Después de eso, las vías de transformación id entre cada par de esquemas unión son también producidas

automáticamente. La integración de esquemas XML en Automed puede efectuarse tanto hacia abajo como hacia arriba. Con el enfoque hacia arriba, el esquema global no está predefinido y es automáticamente generado.

2.3.1 Esquema para la representación de las fuentes de datos XML en HDM

Los documentos XML pueden no tener asociado un DTD o un esquema XML. Estas son gramáticas complejas a las cuales los documentos deben ajustarse. En un escenario de integración de datos, un leguaje de definición de datos de esta complejidad no es necesario. Sin embargo, la estructura del documento es crucial para los algoritmos de integración del esquema y los datos. Por esta razón, Automed introduce el Esquema de Fuente de Datos XML (XMLDSS), el cual es automáticamente derivable de un documento XML, y abstrae sólo su estructura.

Para obtener el XMLDSS de un documento XML, se copia primero en memoria la representación DOM (W3C, 2002) del documento que se modifica para convertirlo en el Esquema de Fuente de Datos.

Como se vio anteriormente HDM es un modelo de datos de bajo nivel que puede representar lenguajes de modelado de alto nivel tales como ER, relacional, orientado a objeto, y XML. Las construcciones de esquemas de alto nivel son identificadas por sus relaciones de objetos. La selección de un modelo de datos común de bajo nivel para Automed fue intencional, de manera que fuera capaz de representar mejor lenguajes de modelado de alto nivel sin incongruencias semánticas o ambigüedades.

Tabla 2.1 Especificación de las construcciones XML en términos de HDM.

Construcción de Alto Nivel	**Representación HDM equivalente**
Construcción elemento Clase nodal Esquema <<e>>	Nodo <<xml:e>>
Construcción atributo Clase enlace-nodal, restricción Esquema <<e,a>>	Nodo <<xml:e:a>> Arista <<_, xml:e, xml:e:a>> Enlaces <<xml:e>> Cons makeCard(<<_,xml:e, xml:e:a>>, 0:1, 1:N)
Construcción lista anidada	Arista <<xml:I, e_p, xml: e_c >>

Clase enlace, restricción Esquema <<e_p, e_c, i>>	Cons makeCard(<< xml:I, e_p, xml: e_c >>, 0:N, 1:1)
Construcción pcdata Clase nodal Esquema <<pcdata>>	Nodo <<xml:pcdata>>

La tabla 2.1 muestra la representación de las construcciones del Esquema de Fuentes de Datos XML en términos de HDM. Los Esquemas de Fuentes de Datos XML constan de cuatro construcciones:

1. Un elemento e puede existir por sí mismo y es una construcción nodal. Se representa por el esquema e.
2. Un atributo a perteneciente a un elemento e es una construcción nodal-enlace. En términos de HDM esto significa que un atributo se representa por un nodo representando el atributo con esquema xml:e:a, una arista relacionando el nodo atributo con su elemento con el esquema _,xml : e,xml : e : a, y una restricción de cardinalidad que declara que una instancia de e puede tener como máximo una instancia de a asociada a ella, mientras que una instancia de a puede estar asociada con una o mas instancias de e.
3. La relación padre-hijo entre dos elementos e_p y e_c es una construcción de enlace con esquema << e_p, e_c, i >>, donde i es el orden de e_c en la lista de hijos de e_p. En términos de HDM, esto se representa por una arista entre e_p y e_c, y una restricción de cardinalidad que declara que cada instancia de e_p se asocia con 0 o mas instancias de e_c, mientras que una instancia de e_c se asocia con exactamente una instancia de e_p .
4. El texto en XML se representa por la construcción PCData. Esta es una construcción nodal con esquema pcdata. En cualquier esquema, solo hay una construcción PCData. Para relacionar la construcción PCData con un elemento esta se trata como un elemento y se usa la construcción lista-anidada.

2.3.2 Transformaciones primitivas en XML

Una vez que las construcciones XML se definen en términos de las construcciones HDM, se puede derivar automáticamente el conjunto necesario de transformaciones primitivas expresado en ese lenguaje. Por tanto, a partir de la

especificación de XML dada en la tabla 2.1, las transformaciones primitivas de la tabla 2.2 se pueden derivar automáticamente.

Tabla 2.2: Transformaciones derivadas sobre XML

Transformación en XML	Transformación equivalente en HDM
$renombrarElem_{XML}$(e,e')	renombrarNodo(<<xml:e>>,<< xml:e'>>)
$addElem_{XML}$(e,q)	addNodo(<<xml:e>>,q)
$renombrarAtrib_{XML}$(a,a')	renombrarNodo(<<xml:e:a>>,<<xml:e:a'>>)

$addAtrib_{XML}$(e,a,q_{Atrib},q_{Asoc})	addNodo(<<xml:e:a>>,q_{Atrib});
	addArista(<<_,xml:e, xml:e:a>>,q_{Asoc});
	addRestriccion(x∈<<xml:e:a>>→<_,x>,∈
	<<_,xml:e, xml:e:a>>)

$delAtrib_{XML}$(e,a,q_{Atrib},q_{Asoc})	delRestriccion(x∈<<xml:e:a>>→<_,x>∈ <<_,xml:e, xml:e:a>>); delArista(<<_,xml:e, xml:e:a>>,q_{Asoc}); delNodo(<<xml:e:a>>,q_{Atrib})
$addLista_{XML}$(<<e,e_S>>,q ,p)	addArista(<<_,xml:e, xml:e_S>>, q,list(p))
$delLista_{XML}$(<<e,e_S>>,q ,p)	delArista(<<_,xml:e, xml:e_S>>, q,list(p))
$addSet_{XML}$(<<e,e_S>>,q)	addArista(<<_,xml:e, xml:e_S>>, q)
$delSet_{XML}$(<<e,e_S>>,q)	delArista(<<_,xml:e, xml:e_S>>, q)

En la Tabla 2.2 la columna de la izquierda muestra los nombres y argumentos de las transformaciones XML y la de la derecha muestra su implementación como secuencias de transformaciones primitivas sobre la representación HDM subyacente.

2.3.3 Algoritmo de transformación de esquema

El algoritmo de transformación de esquema XML se puede aplicar de dos maneras: de arriba hacia abajo, donde el esquema global es predefinido y los esquemas de datos fuentes se transforman para hacerlos corresponder con este, sin tener en cuenta cualquier perdida de información; o de abajo hacia arriba, donde no existe un esquema global predefinido y se preserva la información de todas las fuentes de datos. Ambos enfoques crean las vías de transformación que producen los esquemas intermedios con estructura idéntica. Estos esquemas son transformados entonces en los esquemas unión incluyendo las vías de transformaciones id entre ellos. La vía de transformación de uno de los esquemas unión al esquema global

puede generarse luego de una u otra forma: ya sea automáticamente, usando semánticas '*append*', o semiautomáticamente, en cuyo caso se necesita que las consultas provistas con las transformaciones que especifican la integración sean proporcionadas por el usuario. Por semántica '*append*' nos referimos a que las extensiones de las construcciones del esquema global son creadas añadiendo las extensiones de las construcciones correspondientes de los esquemas unión a su vez. Por tanto, si las fuentes de datos XML se integraran en un orden diferente, la extensión de cada construcción del esquema global contendría las mismas instancias, pero en orden diferente.

Enfoque de abajo hacia arriba

En este enfoque, no está presente un esquema global GS, sino que este es creado automáticamente a partir de las fuentes de datos, sin perdida de información. El algoritmo de transformación de esquemas se aplica a los esquemas de datos fuentes en una forma por grupos de dos, con el fin de derivar incrementalmente el esquema unión compatible de cada uno (figura 2.4). Los esquemas de datos fuentes Si se transforman primero en esquemas intermedios. Luego, se producen los esquemas unión junto con las transformaciones id.

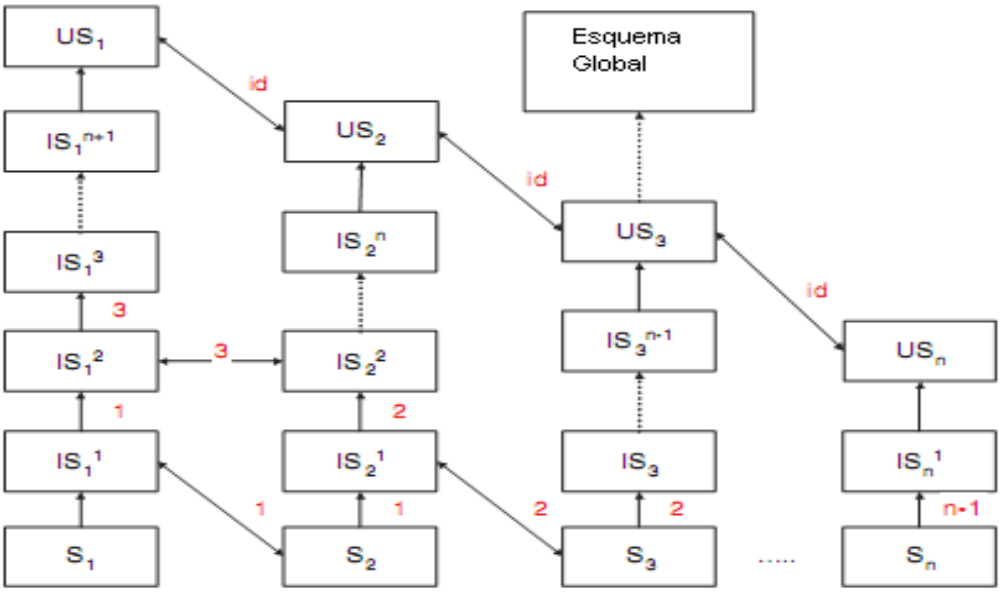

Figura 2.4 Integración de Esquema de Fuente De Datos XML

Para empezar, el esquema intermedio del primer esquema de datos fuente es en sí, $S_1 = IS_1^1$. Entonces se emplea el algoritmo de transformación de esquemas en IS_1^1 y S_2 (ver la anotación 1 en la figura 2.4). El algoritmo complementa IS_1^1 con las

construcciones de S_2 que el no contiene. También reestructura S_2 para hacerle corresponder la estructura de IS_1^1, complementándolo también con las construcciones de IS_1^1 que no contiene. El mismo proceso se lleva a cabo entre IS_2^1 y S_3, resultando en la creación de IS_2^2 y IS_3^1 (anotación 2). El algoritmo se aplica entre IS_1^2 y IS_2^2 resultando solo en la creación de IS_1^3, ya que ahora IS_1^2 no tiene ninguna construcción que IS_2^2 no contenga (anotación 3). Los esquemas intermedios que quedan se generan de la misma forma: para producir el esquema IS_i, el algoritmo de transformación de esquemas es empleado en IS_{i-1}^1 y IS_i, resultando en la creación de IS_{i-1}^2 y IS_i^1, los demás esquemas intermedios excepto IS_{i-1}^2 y IS_i^1 son luego extendidos con las construcciones de LS_i que ellos no contengan. Finalmente generamos automáticamente los esquemas unión US_i, las transformaciones id entre ellos, y el esquema global (usando semántica *append*).

2.3.4 Consulta de archivos XML

Después de la generación de sendas de transformación ya sea por la integración de abajo hacia arriba o de arriba hacia abajo, las consultas enviadas al esquema global pueden ser evaluadas. Este procesamiento de consulta ya fue explicado en la sección 2.2.3.

Las clases AutoMedWrapperFactory y AutoMedWrapper son clases abstractas que implementan algunos de los métodos abstractos, mientras que las clases XMLWrapperFactory y XMLWrapper implementan los métodos abstractos restantes. Las clases Factory tratan con aspectos específicos del modelo, por ejemplo. las llaves primarias para bases de datos relacionales.

2.4 Las API de Automed

Los repositorios API de Automed brindan una interfaz Java para el MDR y el STR. Proporcionan métodos para crear, consultar, alterar y eliminar modelos, construcciones, esquemas de construcciones, esquemas, objetos de esquema, esquemas de objetos, transformaciones así como también algunas entidades auxiliares tales como los métodos de acceso para esquemas extensionales.

La figura 2.5 ilustra como la herramienta y los diferentes repositorios están conectados. El DSR (Repositorio de Fuente de Datos) es una clase auxiliar que provee funcionalidades comunes al MDR y STR.

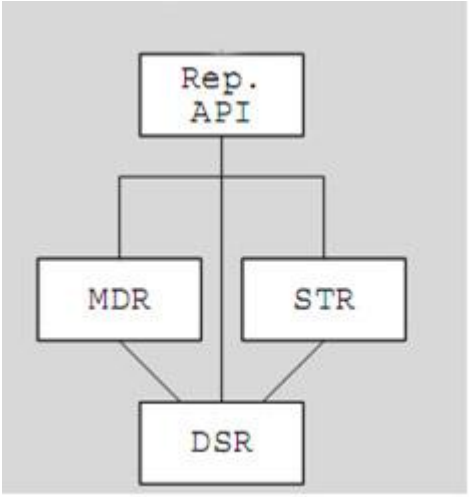

Figura 2.5 Arquitectura de los repositorio API de Automed

2.4.1 Módulo Mediador XML

En Automed, este paso de representación de las fuentes de datos en HDM puede ser formalizado dentro de la metodología de integración de datos si los lenguajes de modelado de datos usados por los esquemas componentes están descritos en el MDR.

Para usar los mediadores de Automed se importan los paquetes de Automed:

Import uk.ac.ic.doc.automed.wrappers.*;

Como estamos tratando con fuentes de datos XML, se importa también el XMLWrapper:

Import uk.ac.bbk.dcs.automed.xml.XMLWrapper;

Las clases principales usadas en este proceso son:

Clase AutoMedWrapper

Un mediador Automed representa una conexión abierta con una fuente de datos. Debido a que las fuentes de datos son tan variadas, muchas de las funcionalidades de un mediador son encargadas a implementaciones de esta clase. Por tanto AutomedWrapper es una clase base para que los mediadores Automed sean utilizados.

La clase XML Wrapper es una clase mediadora para el Repositorio Automed que hereda de AutoMedWrapper. Esta actúa como mediadora entre IQL y los archivos XML en el Repositorio Automed. Esta consulta los archivos XML transformando las consultas IQL dadas en búsquedas XPath, luego transforma los resultados de vuelta a IQL. Un método importante en esta clase es selectNewAutoMedWrapper que, usando el URL y la información del driver, intenta seleccionar el tipo de

mediador adecuado a crear, devolviendo un objeto de tipo AutoMedWrapper. El método getSchemaDoc() retorna el esquema DOM de la fuente de datos XML, sin UID's. El método getSchemaUIDDoc() retorna el esquema DOM de la fuente de datos XML, con UID's. El método getSchemaName() devuelve el nombre del esquema de la fuente de datos XML.

Clase DOMWrapper

Un mediador de bajo nivel que hereda de la clase XMLWrapper. La clase DOMWrapper usa XPath y DOM. El DOMWrapper mantiene en memoria el esquema y los documentos de instancia.

Clase Document:

Define la representación interna para un documento esquema XML.

Clase Schema:

La clase Schema representa un esquema en la base de datos como un conjunto de SchemaObjects. El método getSchemaObject retorna el SchemaObject del esquema que tiene el nombre pasado como argumento. Cada esquema puede ser extensional o intencional.

Clase SchemaObject

La clase SchemaObject tiene el método getSchemaObject que retorna una referencia al SchemaObject cuyo nombre se le pasa como argumento.

2.4.2 Correspondencia, conformación e integración de esquemas

Para llevar a cabo la correspondencia, conformación e integración de esquemas se importa el paquete uk.ac.ic.doc.automed.matching.

Clase SchemaElement

La clase para representar objetos de esquema en la correspondencia de esquemas. El SchemaElement representa un objeto de esquema que puede usarse en el proceso de correspondencia de esquemas. El SchemaElement tiene conocimiento del nombre del objeto de esquema y el tipo de dato de sus instancias. Un SchemaElement es "atómico" cuando tiene tuplas unarias como instancias, en otro caso es "no-atómico".

Clase SemanticRelationship

Esta clase representa la relación semántica entre dos SchemaElement. Las relaciones semánticas son dirigidas, i.e. la relación entre dos elementos A, B dependen de la dirección en la que son comparados. Existen cuatro tipos de

relaciones semánticas: equivalencia, inclusión, solapamiento y disyunción. La inclusión puede ser de subconjunto o súper conjunto. Las relaciones tienen una fortaleza por ejemplo, la relación más fuerte es la de equivalencia y la más débil es la de disyunción.

Clases Disyunción, Equivalencia, Incompatibilidad, Solapamiento e Inclusión.

Estas clases heredan de SemanticRelationship. Representan la relación semántica disyunción, equivalencia, incompatibilidad, solapamiento e inclusión entre dos objetos SchemaElement, respectivamente. El método getRelationship devuelve un objeto SemanticRelationship que es una instancia de esta clase.

Clase SchemaElementPair

Clase que representa un par de SchemaElement

Clase SchemaElementPairRelationship

Esta clase hereda de SchemaElementPair y representa un par de elementos y su relación. A su constructor se le pasa como argumentos el par de SchemaElement y la relación entre estos. El orden de los SchemaElement es importante ya que la relación semántica es dirigida, y su dirección es del primer elemento al segundo elemento del par.

Clase Integrator

La clase que representa el componente Integrador en la arquitectura de la correspondencia de esquemas. Su rol es la integración de los esquemas basado en el descubrimiento de las relaciones entre ellos.

2.4.3 Materialización del esquema global

Para materializar el esquema global obtenido se importa el paquete uk.ac.bbk.dcs.automed.xml.materialisation.

Clase XMLDSSDOMMaterialiser

Esta clase genera un documento con el nombre y en la dirección especificados que representa el esquema global materializado.

2.5 Conclusiones parciales

Se ha presentado la arquitectura de Automed, destacando las funcionalidades de sus componentes y su rol en el proceso de integración. Se hizo un análisis de las etapas de integración de esquemas XML y se describieron las clases principales de las API que proporcionan las operaciones para efectuar dicho proceso.

CAPÍTULO III. **HERRAMIENTA DE INTEGRACIÓN DE DATOS XML**

En este capítulo se describen las características generales de la herramienta así como su relación con las API de Automed y las indicaciones para su uso. Como prueba a la herramienta se realiza el proceso de integración de dos documentos XML heterogéneos. Finalmente se analizan las vulnerabilidades de la herramienta.

3.1 Características generales de la herramienta

La herramienta hace uso de las API de Automed para llevar a cabo el proceso de integración. La clase MóduloXML establece una conexión a la base de datos mediante operaciones definidas en las API, para la representación del esquema XML en HDM y el almacenamiento de todos los datos relacionados con los esquemas a integrar. La clase Correspondencia efectúa una comparación por pares de los elementos de los esquemas existentes en la base de datos para identificar las relaciones entre ellos. En este proceso se generan transformaciones automáticamente, creándose los esquemas intermedios en la base de datos. Estos esquemas se reestructuran y se obtiene el esquema global. En la figura 3.1 se muestra el diagrama de estas clases.

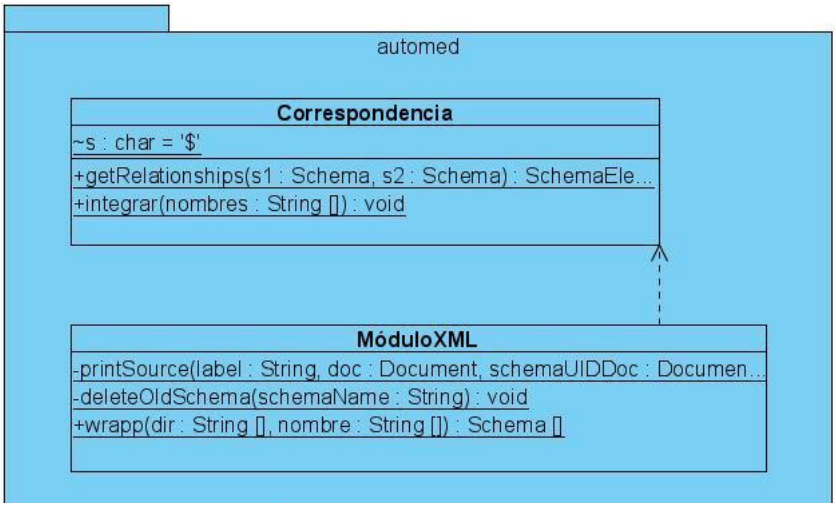

Figura 3.1 Diagrama de clases

3.2 Guía de usuario

Como la herramienta usa el repositorio Automed, el cual esta implementado como un paquete Java, debe estar disponible un ambiente de tiempo de ejecución Java (JRE) 1.4 o de desarrollo, y una cuenta de base de datos Postgre para almacenar los datos del repositorio.

Para configurar el acceso de las API a la base de datos se debe modificar el archivo creado $HOME/.automed/data_source_repository.cfg, donde se puede definir el nombre y el nombre del dominio de la base de datos, así como la contraseña y el nombre de usuario para acceder a esta

La herramienta tiene una interfaz simple de usuario como se muestra en la figura 3.2, con un menú llamado Repositorio que permite definir el repositorio que reside en la base de datos en Postgre, denominada Automed y crear el modelo HDM en ese repositorio, y seleccionar en un directorio los documentos XML a integrar. El botón Integrar inicia el proceso de integración de los documentos seleccionados. En las áreas de texto debajo del botón se muestra la representación en HDM de los esquemas fuente, como resultado de la primera etapa del proceso. Luego de que la integración se haya realizado, se puede visualizar el esquema global pulsando el botón Esquema Global.

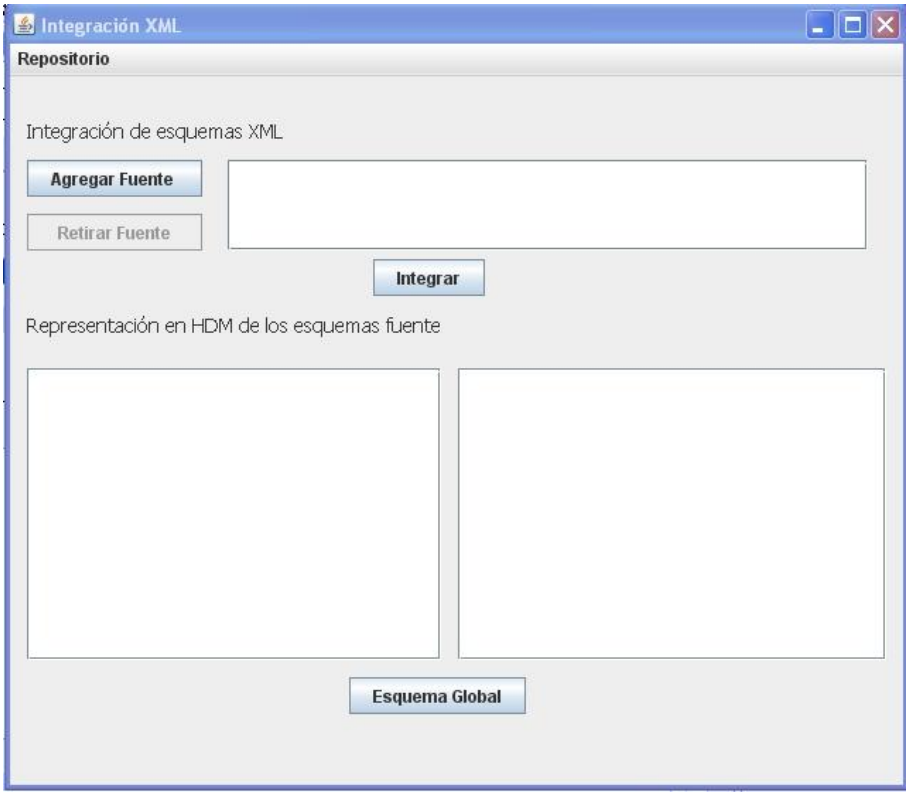

Figura 3.2 Interfaz de usuario

El tratamiento de errores se efectúa para los casos en que el usuario pulse el botón Integrar sin antes agregar a la lista los documentos XML, como se muestra en la figura 3.2, y cuando el usuario intenta ver el esquema global sin antes haber integrado los esquemas fuente (figura 3.3).

Figura 3.3 Mensaje de alerta en el primer caso

Figura 3.4 Mensaje de alerta en el segundo caso

3.3 Pruebas realizadas a la herramienta

En este epígrafe se presenta un caso de estudio para describir el proceso de integración con la herramienta desarrollada. Se tienen dos documentos XML que presentan estructuras heterogéneas, y contienen información sobre libros. En la figura 3.4 se muestran estos dos documentos.

```
<tema tipo='Novela'>
 <autor nombre='U. Eco'>
  <libro>
   <ISBN>0099466031</ISBN>
   <titulo>El nombre de la rosa</titulo>
   <editor>Vintage</editor>
  </libro>
 </autor>
</tema>
```

```
<autor>
 <nombre> N. A. Buddy</nombre>
  <libro>
  <ISBN>0A7B21C6D2</ISBN>
   <título>Principios de la Computación</título>
   <género tipo='Ciencias de la Computacion'></género>
  </libro>
</autor>
```

Figura 3.5 Documentos XML heterogéneos

El primer paso en la integración en Automed es extraer el Esquema de Fuente de Datos del documento XML y hacer su representación en HDM. El resultado de este paso se muestra en la figura 3.6.

schema:L1
text:<<text>>
xmlElement:<<tema$1>>
xmlAttribute:<<_,tema$1,tipo,_>>
xmlElementRel:<<1,tema$1,text,_>>
xmlElement:<<autor$1>>
xmlAttribute:<<_,autor$1,nombre,_>>
xmlElementRel:<<2,tema$1,autor$1,_>>
xmlElementRel:<<1,autor$1,text,_>>
xmlElement:<<libro$1>>
xmlElementRel:<<2,autor$1,libro$1,_>>
xmlElementRel:<<1,libro$1,text,_>>
xmlElement:<<ISBN$1>>
xmlElementRel:<<2,libro$1,ISBN$1,_>>
xmlElementRel:<<1,ISBN$1,text,_>>
xmlElementRel:<<3,libro$1,text,_>>
xmlElement:<<titulo$1>>
xmlElementRel:<<4,libro$1,titulo$1,_>>
xmlElementRel:<<1,titulo$1,text,_>>
xmlElementRel:<<5,libro$1,text,_>>
xmlElement:<<editor$1>>
xmlElementRel:<<6,libro$1,editor$1,_>>
xmlElementRel:<<1,editor$1,text,_>>
xmlElementRel:<<7,libro$1,text,_>>
xmlElementRel:<<3,autor$1,text,_>>
xmlElementRel:<<3,tema$1,text,_>>

schema:L2
text:<<text>>
xmlElement:<<autor$1>>
xmlElementRel:<<1,autor$1,text,_>>
xmlElement:<<nombre$1>>
xmlElementRel:<<2,autor$1,nombre$1,_>>
xmlElementRel:<<1,nombre$1,text,_>>
xmlElementRel:<<3,autor$1,text,_>>
xmlElement:<<libro$1>>
xmlAttribute:<<_,libro$1,ISBN,_>>
xmlElementRel:<<4,autor$1,libro$1,_>>
xmlElementRel:<<1,libro$1,text,_>>
xmlElement:<<titulo$1>>
xmlElementRel:<<2,libro$1,titulo$1,_>>
xmlElementRel:<<1,titulo$1,text,_>>
xmlElementRel:<<3,libro$1,text,_>>
xmlElement:<<editor$1>>
xmlElementRel:<<4,libro$1,editor$1,_>>
xmlElementRel:<<5,libro$1,text,_>>
xmlElement:<<género$1>>
xmlAttribute:<<_,género$1,tipo,_>>
xmlElementRel:<<6,libro$1,género$1,_>>
xmlElementRel:<<7,libro$1,text,_>>
xmlElementRel:<<5,autor$1,text,_>>

Figura 3.6 Representación de las construcciones XML en HDM

La construcción pcdata (text) se trata como un elemento más y se relaciona con los demás elementos mediante la construcción lista anidada. La relación padre-hijo entre dos elementos tiene primero el orden de ese hijo en la lista de hijos del padre. Toda la información referente al modelo de datos, los esquemas, las construcciones, los objetos y sus relaciones, se guarda en la base de datos. En la

tabla Modelo se agrega el nuevo modelo xmldss (figura 3.7) y en la tabla Construcción los tipos de construcción de este modelo (figura 3.8).

mid	name
1	hdm
2	xmldss

Figura 3.7 Tabla Modelo

mid	cid	name	class	is_root
1	1	node	1	1
1	2	edge	3	1
1	3	nodeOrEdg	5	0
1	4	constraint	4	1
2	5	xmlElement	1	1
2	6	text	1	1
2	7	xmlAttribute	2	1
2	8	xmlElement	5	0
2	9	xmlElement	3	1

Figura 3.8 Tabla Construcción

El segundo paso consiste en identificar las relaciones semánticas entre los elementos de los esquemas, y a partir de estas se generan automáticamente las transformaciones, creando los esquemas intermedios.

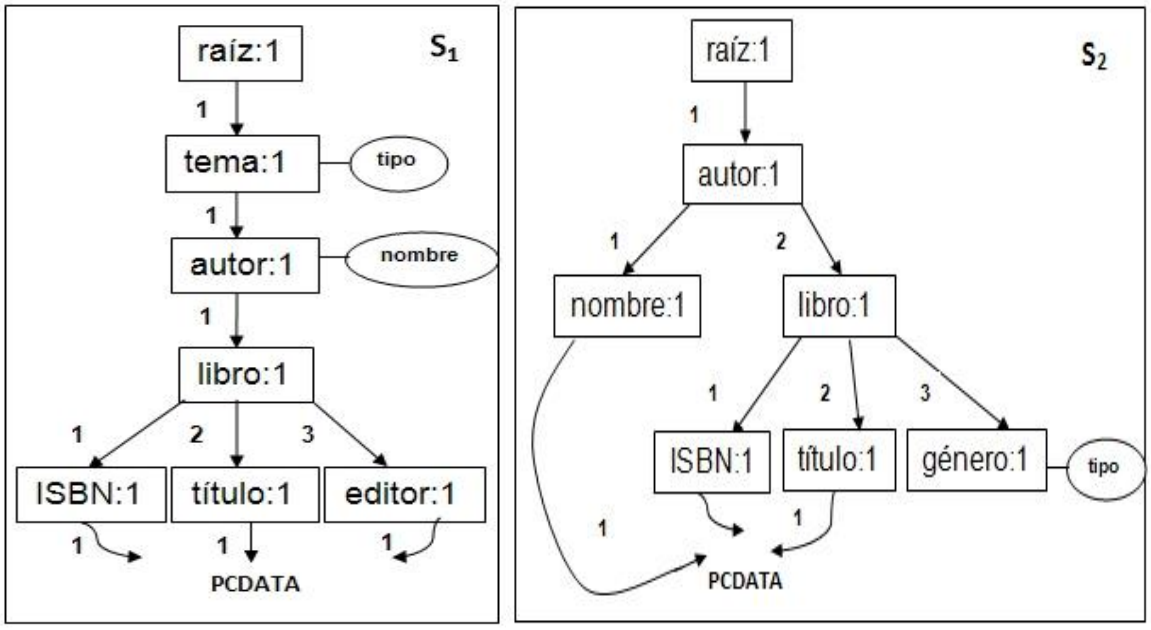

Figura 3.9 Ejemplo de esquemas

En este caso el proceso de correspondencia de esquemas deberá descubrir que el elemento tema en S_1 se corresponde con el elemento género en S_2 y generará la transformación:

renombrarElemento(<<tema:1>>,<<género:1>>)

En la tabla Esquema se almacenan los datos de estos esquemas intermedios creados (**Error! Reference source not found.** y en la tabla Transformación (figura 3.11) la transformación realizada.

sid	name	type	subnet_sid
1	L1	3	1
2	L2	3	2
3	L1a	0	1

Figura 3.10 Esquema intermedio resultado de la transformación

action	from_ob	to_obj	from_schema	to_schema	function	cons_pred
rename	96	97	1	3	Null	True

Figura 3.11 Transformación de renombre

Posteriormente se realiza la restructuración de los esquemas intermedios aplicándose el algoritmo de restructuración. Las transformaciones realizadas en S_1 serian:

1. para agregar el elemento nombre y eliminar el atributo nombre:
   extendNodo(<<nombre:1>>, q)
   extendArista(<<autor:1>>, <<nombre:1>>, q)
   extendArista(<<nombre:1, PCDATA>>, q)
   eliminarArista(<<autor:1, autor:nombre>>, q)

En la figura 3.12 se muestran las transformaciones efectuadas.

action	from_ob	to_obj	from_schema	to_schema	function	cons_pred
extend	Null	87	1	3	Range Void	True
extend	Null	88	3	4	Range Void	True
extend	Null	90	4	5	Range Void	True
delete	9	Null	5	6	[{a}]{a}<-<<aut	True

Figura 3.12 Transformación de atributo-elemento

2. para colocar el elemento género como hijo del elemento libro:
   extendArista(<<libro>>,<<género>>, q)

contractArista(<<género>>,<<autor>>, q)

Que genera las transformaciones de la figura 3.13.

extend	Null	127	18	19	[(b,g)](g,a)<-<	True
contract	114	Null	19	20	[(g,a)](a,b)<-<	True

Figura 3.13 Transformación de arista

Las transformaciones efectuadas en S_2 serian:

1. Agregar elemento editor

   extendNodo(<<editor:1>>, q)

   extendArista(<<libro:1>><<editor:1>>,q)

   extendArista(<<editor:1, PCDATA>>, q)

extend	Null	108	12	13	Range Void	True
extend	Null	110	13	14	Range Void	True

Figura 3.14 Transformación de elemento

Luego de la restructuración se efectuó una transformación de tipo *ident* que hace corresponder dos construcciones sintácticamente idénticas en dos esquemas intermedios diferentes. Finalmente se selecciona uno de los esquemas a los que se le aplicó la transformación *ident* como esquema global.

En la figura 3.15 se muestra el esquema global obtenido.

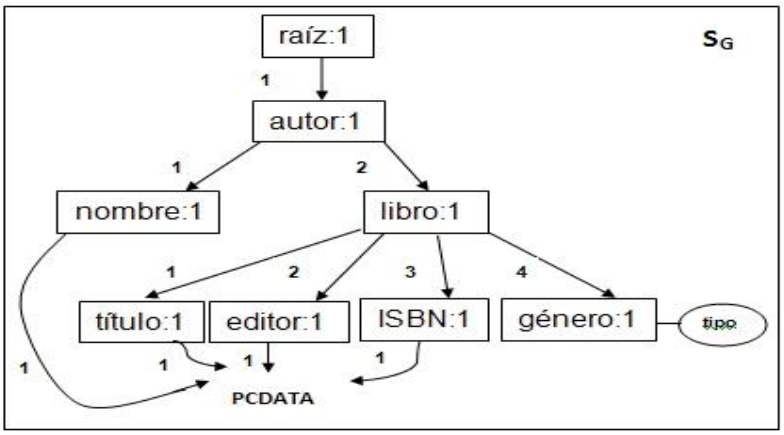

Figura 3.15 Esquema global

3.4 Vulnerabilidades de la herramienta

Entre las vulnerabilidades identificadas en la herramienta se encuentra que en la correspondencia de esquemas no se usa conocimiento externo de ontologías o diccionarios, ya que el proceso de comparación léxica se hace mediante los grados de similitud entre los elementos. Esto limita la capacidad del sistema para utilizarlo

en dominios específicos. Este proceso es semiautomático, de modo que se requiere especificar los nombres de los elementos de los esquemas y atendiendo a sus características especificas realizar la identificación de las relaciones entre los elementos.

Otra limitación es el uso de un lenguaje simple de modelado llamado *Esquema de Fuente de Datos* XML (XMLDSS) para extraer la estructura de un documento XML y llevarla a representación HDM, en lugar de obtener esta estructura de los lenguajes de definición de esquema como DTD's o Esquemas XML.

3.5 Conclusiones parciales

Se han descrito las características principales de la herramienta y las indicaciones para su uso. Como prueba se efectuó la integración de dos documentos XML heterogéneos, describiéndose los resultados por etapas y la información del proceso almacenada en la base de datos. Además se analizaron las limitaciones de la metodología utilizada para la creación de la herramienta.

CONCLUSIONES

Se realizó un estudio de los sistemas de integración de datos reportados en la bibliografía, específicamente de los sistemas de integración de datos XML, haciendo énfasis en el sistema Automed cuya metodología, más ventajosa que las propuestas en otros proyectos, fue adoptada para la creación de una herramienta que integra datos XML en un esquema global.

RECOMENDACIONES

- Incorporar la herramienta desarrollada a un Sistema de Recuperación de Información sobre documentos XML heterogéneos.
- En el proceso de correspondencia de esquemas debe implementarse un módulo *nombre* más inteligente utilizando conocimiento externo de diccionarios u ontologías.

BIBLIOGRAFÍA

1. ABITEBOUL, S., MANOLESCU, I., NGUYEN, B. & PREDA, N. (2005) A Test Platform for the INEX Heterogeneous Track. IN FUHR, N., LALMAS, M., MALIK, S. & SZLÁVIK, Z. (Eds.) *Advances in XML Information Retrieval.* Springer Berlin / Heidelberg.
2. ADALI, S., CANDAN, K. S., PAPAKONSTANTINOU, Y. & SUBRAHMANIAN, V. S. (1996) Query caching and optimization in distributed mediator systems. *Proc. of ACM SGMOD.*
3. ARENS, Y., KNOBLOCK, C. A. & HSU, C. (1996) Query processing in the SIMS information mediator. *Advanced Planning Technology.* Menlo Park, CA, AAAI Press.
4. AZEVEDO, M., PAIXÃO, K. & PEREIRA, D. (2006) Processing Heterogeneous Collections in XML Information Retrieval. IN FUHR, N., LALMAS, M., MALIK, S. & KAZAI, G. (Eds.) *Advances in XML Information Retrieval and Evaluation.* Springer Berlin / Heidelberg.
5. BATINI, C., LENZERINI, M. & NAVATHE, S. (1986) A comparative analysis of methodologies for database schema integration. *ACM Computing Surveys,* 18(4).
6. BERNERS-LEE, T., HENDLER, J. & LASSILA, O. (2001) The semantic web. *Sci Am,* 284 (5).
7. BERNSTEIN, P. (2003) Applying Model Management to Classical Meta Data Problems *CIDR*.
8. BERNSTEIN, P. & RAHAM, E. (2000) Data warehouse scenarios for model management. *Proc 19th Int Conf On Entity-Relationship Modeling.* New York.
9. BOUZEGHOUB, M. & LENZERINI, M. (2001) Introduction to the special issue on data extraction, cleaning, and reconciliation. *Information Systems,* 26(8).
10. BOYD, M., KITTIVORAVITKUL, S., LAZANITIS, C., MCBRIEN, P. J. & RIZOPOULOS, N. (2004) Automed: A BAV data integration system for heterogeneous data sources. *Proc. CAiSE2004,* 3084 of LNCS.
11. BUNEMAN, P. (1997) Semistructured data. *Proceedings of the Symposium on Principles of Database Systems PODS-97.* Tucson, Arizona.

12. CAREY, M. J., HAAS, L. M., SCHWARZ, P. M., ARYA, M., CODY, W. F., FAGIN, R., FLICKNER, M., LUNIEWSKI, A. W., NIBLACK, W., PEKTOVIC, D., THOMAS, J., WILLIAMS, J. H. & WIMMERS, E. L. (1994) Towards Multimedia Information System: The Garlic Approach. *IBM Almaden Research Center.* San Jose.
13. CHAWATHE, S. S., GARCIA-MOLINA, H., HAMMER, J., IRELAND, K., PAPAKONSTANTINOU, Y., ULLMAN, J. D. & WIDOM, J. (1994) The TSIMMIS Project: Integration of Heterogeneous Information Sources. *IPSJ.*
14. DOMENIG, R. & DITTRICH, K. (1999) An Overview and Classification of Mediated Query Systems. *SIG-MOD Record,* 28 (3).
15. DRAGONI, A. F. (1997) Belief revision: from theory to practice. *The Knowledge Engineering Review*
16. FROMMHOLZ, I. & LARSON, R. (2007) The Heterogeneous Collection Track at INEX 2006. IN FUHR, N., LALMAS, M. & TROTMAN, A. (Eds.) *Comparative Evaluation of XML Information Retrieval Systems.* Springer Berlin / Heidelberg.
17. HALEVY, A. Y. (2001) Answering queries using views: A survey. *Very Large Database J.,* 10 (4).
18. HAMMER, J., GARCÍA-MOLINA, H., NESTOROV, S., YERNENI, R., BREUNIG, M. & VASSALOS, V. (1997) Template-based wrappers in the TSIMMIS system. *Proceedings of the 1997 ACM SIGMOD international conference on Management of data.* Tucson, Arizona, United States, ACM.
19. HULL, R. (1997) Managing semantic heterogeneity in databases: A theoretical perspective. *Proc. of the 16th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems.*
20. LARSON, R. (2007) Probabilistic Retrieval Approaches for Thorough and Heterogeneous XML Retrieval. IN FUHR, N., LALMAS, M. & TROTMAN, A. (Eds.) *Comparative Evaluation of XML Information Retrieval Systems.* Springer Berlin / Heidelberg.
21. LEHTONEN, M. (2005) EXTIRP 2004: Towards Heterogeneity. IN FUHR, N., LALMAS, M., MALIK, S. & SZLÁVIK, Z. (Eds.) *Advances in XML Information Retrieval.* Springer Berlin / Heidelberg.

22. LENZERINI, M. (2002) Data integration: a theoretical perspective. *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems.* Madison, Wisconsin, ACM.
23. LESSER, V., HORLING, B., KLASSNER, F., RAJA, A., WAGNER, T. & S. XQ, Z. (1998) BIG: A resource-bounded information gathering agent. *Proceedings of the AAAI-98.* Madison, Wisconsin.
24. MADHAVAN, J., BERNSTEIN, P. & RAHM, E. (2001) Generic schema matching with Cupid. *Proceedings of the Very Large Data Bases Conference (VLDB).*
25. MCBRIEN, P. (2003) Data Integration by Bi-Directional Schema Transformation Rules. IN ALEXANDRA, P. (Ed.
26. MCBRIEN, P. J., JASPER, E., TONG, N. & POULOVASSILIS, A. (2003) View generation and optimization in Automed data integration framework. *Automed Project.*
27. MENA, E., KASHYAP, V., SHETH, A. & ILLARRAMENDI, A. (1996) OBSERVER: An approach for query processing in global information systems based on interoperation across pre-existing ontologies. *Proceedings of the International Conference on Cooperative Information Systems CoopIS-96.* Brussels, Belgium.
28. OGILVIE, P. & CALLAN, J. (2006) Parameter Estimation for a Simple Hierarchical Generative Model for XML Retrieval. IN FUHR, N., LALMAS, M., MALIK, S. & KAZAI, G. (Eds.) *Advances in XML Information Retrieval and Evaluation.* Springer Berlin / Heidelberg.
29. PALOPOLI, L., SACCA, D. & URSINO, D. (1998) Automatic derivation of terminological properties from database schemes. *Proceedings of DEXA98.* Wien, Austria.
30. PASSI, K., LANE, L., MADRIA, S., SAKAMURI, B., MOHANIA, M. & BHOWMICK, S. (2002) A Model for XML Schema Integration. IN BAUKNECHT, K., TJOA, A. & QUIRCHMAYR, G. (Eds.) *E-Commerce and Web Technologies.* Springer Berlin / Heidelberg.
31. POPA, L., VELEGRAKIS, Y., MILLER, R. J., HERNANDEZ, M. A. & FAGIN, R. (2002) Translating Web Data. *Proc. VLDB-02.*

32. POULOVASSILIS, A. (2001) The Automed Intermediate Query Language. *Automed Project.*

33. POULOVASSILIS, A. & MCBRIEN, P. J. (1998) A general formal framework for schema transformation. *Data Knowl. Eng.,* 28.

34. REYNAUD, C., SIROT, J. P. & VODISLAV, D. (2001) Semantic Integration of XML Heterogeneous Data Sources. *Proc. IDEAS*.

35. RIZOPOULOS, N. (2003) Discovery of semantic relationships between schema elements.

36. RODRIGUEZ-GIANOLLI, P. & MYLOPOULOS, J. (2001) A semantic approach to XML-based data integration. *Proc. ER-01*.

37. ROTH, M. T. & SHWARTZ, P. (1997) Don't scrap it, wrap it! *Proc. of the 23rd Intl. Conf. on Very Large Databases.*

38. SAUVAGNAT, K. & BOUGHANEM, M. (2005) Using a Relevance Propagation Method for Adhoc and Heterogeneous Tracks at INEX 2004. IN FUHR, N., LALMAS, M., MALIK, S. & SZLÁVIK, Z. (Eds.) *Advances in XML Information Retrieval.* Springer Berlin / Heidelberg.

39. SHETH, A. P. & LARSON, J. A. (1990) Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.,* 22**,** 183-236.

40. SHVAIKO, P. & EUZENAT, J. (2005) A survey of schema-based matching approaches. *Journal on Data Semantics,* IV.

41. SZLÁVIK, Z. & RÖLLEKE, T. (2005) Building and Experimenting with a Heterogeneous Collection. IN FUHR, N., LALMAS, M., MALIK, S. & SZLÁVIK, Z. (Eds.) *Advances in XML Information Retrieval.* Springer Berlin / Heidelberg.

42. ULLMAN, J. D. (1997) Information integration using logical views. *Proc. of the 6th Int. Conf. on Database Theory.*

43. W3C (1998) Guide to the W3C XML Specification (“XMLspec”) DTD

44. http://www.w3.org/XML/1998/06/xmlspec-report-v21.htm#AEN56.

45. W3C (2001) XML Schema Specification .http://www.w3c.org/TR/xmlschema-0,

46. http://www.w3c.org/TR/xmlschema-1,http://www.w3c.org/TR/xmlschema-2.

47. W3C (2002) Document Object Model (DOM) .http://www.w3.org/DOM/.

48. WIEDERHOLD, G. (1992) Mediators in the Architecture of Future Information Systems. *Computer,* 25**,** 38-49.