Detección de Elementos Redundantes en Arquitecturas de Información: Un Enfoque Apoyado en Alineación de Ontologías

Camilo Castellanos[1]

Department of Systems and Computing Engineering
University of Los Andes
Bogotá, Colombia

Dario Correal[2]

Department of Systems and Computing Engineering
University of Los Andes
Bogotá, Colombia

Francisco Murcia[3]

Department of Systems and Computing Engineering
University of Los Andes
Bogotá, Colombia

Abstract

This paper proposes an approach for the detection of potentially redundant elements within Enterprise Architectures (EA). Unintended or unidentified redundancies affect the data quality, promote duplication and may lead to inconsistencies. Evaluating and identifying manually redundant elements in large information systems is a tedious, error-prone, and time-consuming task. Untimely, incomplete or inaccurate evaluations, can affect the dynamics and flexibility organizational that promotes an EA. Our proposal is MDA (Model-Driven Architecture) and Ontology-Alignment based. The main idea is to perform a transformation from an information architecture model toward an OWL (Ontology Web Language) ontology, and to apply alignment techniques to infer correspondences between ontology concepts. Our intention in this paper is to support the information architecture, warning to architect about possible overlapping of entity, attribute and association, in order to support assessment, migration, refinement, integration and management processes of enterprise data schemes.

Keywords: Enterprise Architecture, Information Architecture, MDA, Ontology Matching, Ontology Alignment, Redundancy Detection.

1 Introducción

La Arquitectura Empresarial (AE) es uno de los escenarios mas importantes para abordar la gestión de Sistemas de Información (SI). La definición de una AE mejora y facilita la administración de SI dentro de la organización [24]. La información es un activo estratégico para tomar decisiones ágil y acertadamente dentro del entorno cada vez mas cambiante en el que compiten las organizaciones en la actualidad. El nivel de veracidad, fácilidad y rapidez para acceder a esta información puede definir la continuidad de las organizaciones en mercados altamente competitivos. Para lograr tal dinamismo, es necesario representar formal, integral y claramente las distintas dimensiones organizacionales. Uno de los dominios que conforman la AE, es el de información o datos. Las definiciones en este dominio deben posibilitar, entre otros, el análisis sobre los activos de información de la empresa, para detectar y/o eliminar redundancias no planeadas [4].

Uno de los objetivos centrales del diseño de bases de datos es evitar redundancias no intencionadas [30]. Las causas de diseños con tales redundancias son diversas y van desde estrategias para mejorar el desempeño, la disponibilidad o la seguridad del SI, hasta la falta de comunicación y cooperación al interior de las organizaciones, originando *islas de información* [16][21][18]. Por tanto, se puede diferenciar una redundancia controlada, intencionada y justificada, y otra no planeada que conduce a problemas de desempeño, seguridad, sobrecostos e inconsistencias [16].

1.1 Descripción del Problema

Desde una perspectiva organizacional, la información redundante puede conducir a inconsistencias, solapamiento de entidades, y duplicación de esfuerzos en diseño y desarrollo [16]. Consideremos por ejemplo, un caso típico de elementos redundantes entre diferentes SI de una misma organización. En un esquema S1 existe una entidad *Persona*. Esta entidad tiene los atributos *Nombres* y *Apellidos*. Por otro lado, un esquema S2 con la tabla *Usuario*, la cual posee un campo *NombreCompleto*. Estas redundancias deben ser identificadas en orden de adelantar, por ejemplo, proyectos de integración, adquisición, migración o evolución de SI en el marco de una AE. Una estrategia para identificar tales redundancias es identificar y cuantificar las similitudes entre

[1] Email: cc.castellanos87@uniandes.edu.co
[2] Email: dcorreal@uniandes.edu.co
[3] Email: fa.murcia68@uniandes.edu.co

elementos de diferentes esquemas, este proceso de comparación es conocido como: *Alineamiento de esquemas*.

Un procedimiento manual de alineación implica una alta probabilidad de error y gran inversión de tiempo y recursos [25], debido a los errores, omisiones o demoras en la identificación de tales situaciones. Varios estudios [19][20][25] abordan el problema de encontrar similitudes dentro de esquemas heterogéneos desde la perspectiva del alineamiento o matching de ontologías, teniendo en cuenta las coincidencias que se pueden encontrar entre ontologías y esquemas de bases de datos [27]. Recientes sistemas de alineación automática de ontologías, han sido propuestos alcanzando importantes resultados [20][28][17][14][5]. Sin embargo, estas propuestas no incluyen dentro de su alcance una integración con los artefactos que componen la AE. Por ejemplo, estos sistemas no incluyen las relaciones de los datos con procesos, aplicaciones y servicios, por tanto ofrecen una visión parcial de la AE. Análisis que no se soportan en vistas integrales de la organización, pueden promover decisiones equivocadas.

Desde el punto de vista técnico, automatizar la tarea de alineamiento no es un problema trivial, dado que la semántica exacta del modelo es sólo entendida completamente por sus diseñadores, y no puede ser expresada totalmente por el esquema en si mismo [19]. El alineamiento de esquemas es una tarea que presenta muchos retos, principalmente en dos dimensiones. La primera, es el tamaño de los esquemas a comparar y la frecuencia con que se requieren realizar estas comparaciones. Este es un problema en aumento dado el rápido crecimiento en la cantidad y extensión de fuentes de datos, tanto en SI web y como en comercio electrónico, que requieren integrarse [25]. La segunda, es la complejidad que implica comparar fuentes de datos heterogéneas. Esta heterogeneidad puede ser sintáctica, semántica o terminológica [10].

1.2 Objetivos y Contribuciones

Los principales objetivos de nuestra propuesta son: i) Agilizar y mejorar el proceso de detección de entidades redundantes en el marco de una AE, explotando técnicas de alineamiento de ontologías. ii) Enriquecer el dominio de información de una AE adicionando mapeos que incluyan estas redundancias. iii) Proveer una vista que describa integralmente las diferentes relaciones, explícitas e implícitas, contenidas en complejas AEs.

Como principales contribuciones de este trabajo podemos señalar las siguientes: Extendemos un metamodelo de AE para expresar elementos del dominio de información y las relaciones explícitas e implíticas que existen entre ellas. Desarrollamos una herramienta que automáticamente evalúa un conjunto de ontologías por medio de una herramienta de alineación automática para facilitar y apoyar la labor del arquitecto empresarial en busca de enti-

dades redundantes.

1.3 Estructura del Documento

El resto de este documento esta organizado de la siguiente manera: La Sección 2 ofrece un caso de estudio para motivar nuestra aproximación. La Sección 3 describe el contexto de AE y MDA. La Sección 4 detalla los conceptos de ontología y alineación de ontologías. La Sección 5 desarrolla la propuesta de solución. El proceso de experimentación se expone en la Sección 6. La Sección 7 presenta trabajo relacionado. Y finalmente la Sección 8 reporta las conclusiones.

2 Ejemplo de Motivación

Con el propósito de describir mejor el problema, vamos a tomar dos SI de una organización de prueba utilizada en el Laboratorio de Ingeniería de Software de la Universidad de Los Andes. La Figura 1 muestra segmentos de los dos esquemas involucrados: El de la izquierda (*S1*), corresponde al sistema de inventario y el de la derecha (*S2*) es un sistema para la de gestión de clientes (CRM). Estos diagramas son artefactos que hacen parte de la descripción de la arquitectura de información actual de la empresa. En el escenario de procesos de integración, sincronización y evolución en el marco AE, que involucren estos SI, es clave identificar y analizar solapamiento o redundancias de datos. En la figura están identificadas algunas coincidencias entre

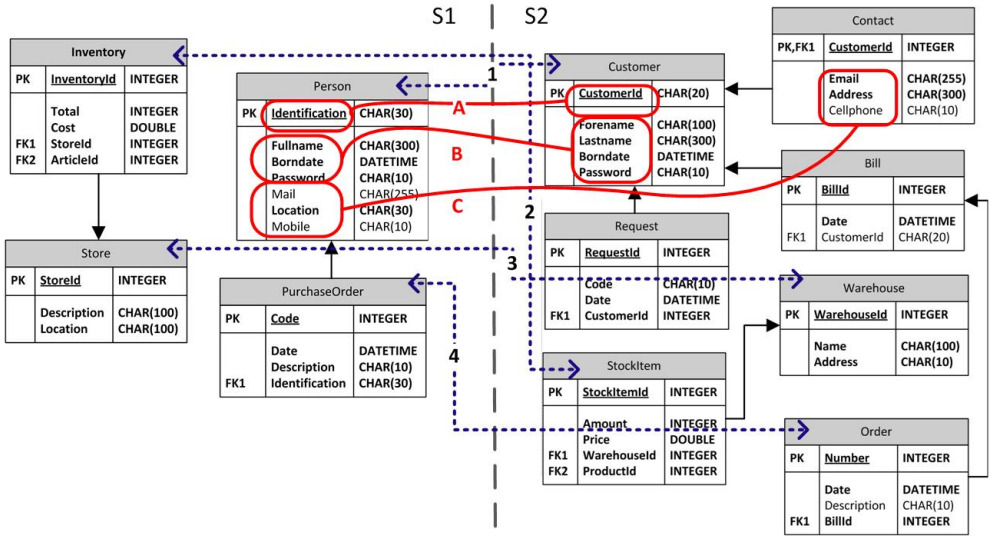

Figura 1. Ejemplo de mapeo entre dos esquemas de una organización

tablas (líneas punteadas) y campos (líneas continuas). Existen corresponden-

Elemento S1	Elemento S2	Sim	Relación
Person	Customer	0.9	=
Store	Warehouse	0.75	=
Person.Fullname	Customer.Forename	0.8	>
Person.Fullname	Customer.Lastname	0.8	>

Tabla 1
Posible Alineamiento entre los esquemas S1 y S2

cias evidentes, como es el caso de $Person.Borndate \simeq Customer.Borndate$ y $Person.Password \simeq Customer.Password$ (*mapeo B*). Otras consideraciones deben tenerse en cuenta, cuando la correspondencia entre los elementos no es 1:1, sino se involucran diferentes cardinalidades, como 1:N, N:1 o M:N. El mapeo: $Person.Fullname \simeq (Customer.Forename + Customer.Lastname)$ da cuenta de ello. En la Sección 4.1 se enumeran algunas técnicas para implementar alineación automática de ontologías. Una vez realizada la confrontación por parte del arquitecto, el mapeo de salida con las coincidencias detectadas es añadido al conjunto de artefactos de la AE. Un posible mapeo resultante de los esquemas descritos en la Figura 1, se puede ver en la Tabla 1. El tipo de relación identificada con '=' denota igualdad, '>' generalidad y '<' especialización entre los elementos de $S1$ y $S2$ respectivamente.

Consideremos ahora, la llegada de un nuevo sistema de Planificación de Recursos Empresariales (ERP) con su propio esquema (*S3*). No sería extraño encontrar elementos comunes entre los tres esquemas. Por tanto, se deben realizar todas las comparaciones con los elementos del nuevo esquema para cubrir las posibles coincidencias. Note que el primer mapeo (S1xS2) contiene información que podrá aprovecharse para inferir nuevas correspondencias. Por ejemplo, si en *S3* existe una entidad *S3.Client*, bastaría con identificar una correspondencia $S1.Person \simeq S3.Client$ para derivar por transitividad, también $S2.Customer \simeq S3.Client$, gracias al mapeo previo $S1.Person \simeq S2.Customer$.

3 Arquitectura Empresarial Dirigida por Modelos

Una AE ofrece una integral y estructurada manera de describir una organización, sus SI, y la forma en que estos se integran a fin de alcanzar los objetivos de negocio apoyados en Tecnologías de Información (IT). Esta descripción se compone de documentos, diagramas y demás artefactos que formalizan diferentes puntos de vista de la organización, de tal manera que sean un referente y soporte para la toma de decisiones. En la actualidad existen numerosos marcos de referencia de AEs entre ellos: Zachman [31], Department Of Defense Architecture Framework (DoDAF) [13], The Open Group Architecture Framework (TOGAF) [29]. Estos frameworks tienen en común la desagregación en

dimensiones de AE: i) La *Arquitectura de Negocio* define la estrategia, gobernabilidad, organización y procesos claves de negocio. ii) La *Arquitectura de Datos* describe la estructura de los activos de datos lógicos y físicos de la organización y los recursos de gestión de datos. iii) La *Arquitectura de Aplicaciones* provee un modelo para las aplicaciones a ser desplegadas, sus interacciones y sus relaciones con los principales procesos de negocio de la organización. iv) En la *Arquitectura de Tecnología* se describen las capacidades de software y hardware que son requeridas para el despliegue de servicios de negocio, datos y aplicaciones.

El dominio de arquitectura de datos es el foco de nuestra contribución y tiene como objetivo definir tipos y fuentes de datos necesarios para soportar el negocio, de manera que sean: Completos, consistentes, estables y entendibles por los usuarios [29].

3.1 Tartarus: Un Metamodelo de Arquitectura Empresarial

Model-Driven Architecture (MDA) es una propuesta de la OMG para abordar el desarrollo de software proporcionando un conjunto de guías para estructurar especificaciones expresadas en modelos. Es neutral en cuanto a tecnología y proveedor, y busca reducir significativamente el esfuerzo de desarrollo, separando la arquitectura del sistema, de las arquitecturas de plataforma. Uno de los elementos claves de MDA es el Modelo de Plataforma Independiente (*PIM*) que describe la estructura y el comportamiento de un sistema, pero no su implementación. La implementanción en la plataforma particular (JEE, .NET, WS, etc) está definida en un Modelo de Plataforma Específica (*PSM*), el cual es originado a partir del PIM. Para materializar esta conversión, se realizan transformaciones basadas en plantillas detalladas para cada plataforma, que mapean elementos del PIM hacia elementos PSM.

Tartarus es un acercamiento MDA para el análisis de AEs del Proyecto *Moosas* de la Universidad de los Andes [22]. Su núcleo es un metamodelo que permite la definición de modelos de AE. Tartarus surge como una opción de solución ante la actual variedad de frameworks, estándares, herramientas y formatos que hacen parte de la definción de una AE [26]. Una primera fase de este proyecto se enfocó en encontrar diferencias entre procesos de negocio.

La Figura 2A ofrece una vista general. El proyecto esta conformado por cinco paquetes: *Enterprise* contiene la estructura, cadena de valor, principios, incentivos organizacionales y demás elementos estratégicos. *Continuum* reúne las definiciones para describir la manera en que la AE evoluciona. *Management* tiene los factores necesarios para evaluar los artefactos que conforman una arquitectura. *Environment* comprende el conjunto de elementos que describen el entorno en el que funciona la empresa. *Architecture* agrupa los conceptos clave para visualizar y estructurar la AE. Este paquete se divide en 3 dominios:

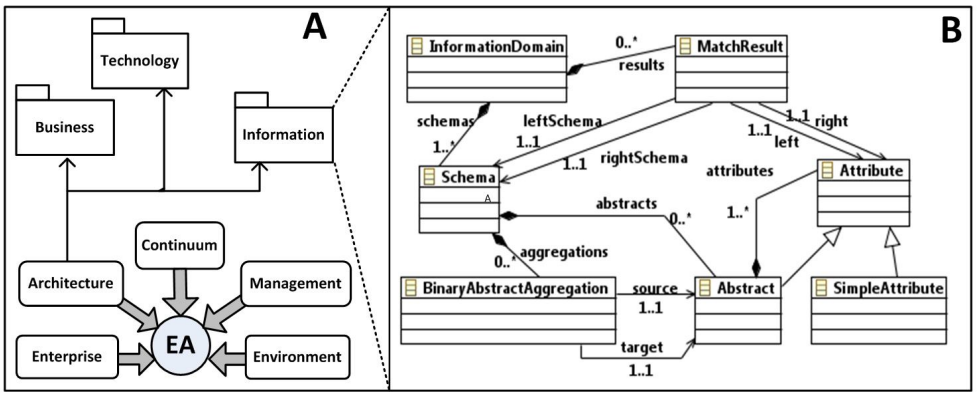

Figura 2. A) Vista General de Tartarus B) Metamodelo del Dominio de Información

Business Domain: Describe los procesos de negocio. *Technology Domain*: Comprende las capacidades de software y hardware que soportan los servicios de negocio e información. *Information Domain*: Estructura los componentes de datos que conforman la información de la empresa.

4 Ontologías y Alineación de Ontologías

Una ontología, básicamente, es una descripción explícita de un dominio de conocimiento específico, definida en términos de sus conceptos, propiedades, atributos, restricciones e individuos [23]. Formalmente podemos definir una ontología como: $O = \{C, P, H^C, H^P, A^O, I, R^I\}$. Donde C es el conjunto de conceptos, P el conjunto de propiedades. H^C es la jerarquía de relaciones entre los conceptos tal que $H^C \subset C \times C(c_i, c_j) \in H^C$ denota que el concepto c_i es subconcepto de c_j. De la misma manera H^P define las relaciones jerárquicas entre propiedades. A^O es el conjunto de axiomas. I comprende el conjunto de Individuos, es decir, instancias de conceptos y propiedades quienes se asocian a través de instancias relacionales R^I. Una de las principales ventajas de las ontologías, es proveer características útiles para sistemas inteligentes, representación e ingeniería de conocimiento [12].

4.1 Alineación de Ontologías.

Una función de alineación de ontoloías puede definirse formalmente así: $f(O_1, O_2) = \{e_{i1}, e_{i2}, i_i, r_i\}$ [10] [11].

Donde O_1 y O_2 son los esquemas/ontologías de entrada, comúnmente llamados origen y destino respectivamente, e_{i1} y e_{i2} son las dos entidades comparadas, i_i corresponde al índice de similitud o confianza (medido entre 0 y 1) y r_i la relación (igualdad, especialización, generalización) que puede haber entre e_{i1} y e_{i2}. Detectar elementos similares entre diferentes fuentes de información es también una necesidad central en procesos de evaluación, migración,

integración y evolución de SI, intercambio de información en sistemas P2P y composición de web services[10].

En 2004 surge la Ontology Alignment Evaluation Initiative (*OAEI*), una iniciativa que anualmente evalúa sistemas de alineamiento de ontologías. El objetivo de la OAEI es comparar diferentes propuestas, con el objetivo de ofrecer conclusiones sobre las mejores técnicas y estrategias, para lo cual, provee unos casos de prueba sobre los cuales los diferentes sistemas experimentan. Entre los temas evaluados se encuentran: *The benchmark track, The directories and thesauri track, Instance matching*. El desempeño en estas pruebas es medido típicamente bajo tres indicadores: La precisión (*Precision*) se refiere a la fidelidad y da cuenta de la proporción de elementos efectivos sobre el total de elementos recuperados. La exhaustividad (Recall), esta asociada con la completitud, y mide la razón entre el total de elementos recuperados sobre el total de elementos que debieron ser identificados. La media armónica (F-Measure), combina tanto precision como recall para determinar un balance entre ellos y oscila entre 0 y 1: $F - Measure = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$.

Existen diferentes métodos para realizar alineamiento automático de ontologías[20][25][27] y nuestra propuesta incluye algunos de ellos. Las principales técnicas de alineamiento son *basadas en esquema*, *basadas en contenido* y *combinadas*. Las *basadas en esquema* sólo tienen en cuenta la información estructural del esquema, no su contenido. Dentro de este grupo se aplican comparaciones lingüísticas, textuales, de restricciones y estructurales. Las estrategias *basadas en contenido* involucran estadísticas, patrones o incluso los mismos datos para inferir correspondencias. Las técnicas *combinadas* aplican, en conjunto, las anteriores aproximaciones en busca de mejores resultados. Esta combinación puede ser configurada manual o automáticamente utilizando aprendizaje de máquina. Este trabajo no busca determinar la mejor forma de detectar posibles redundancias, sino adaptar y aplicar técnicas y avances en el área de alineamiento de ontologías al marco de una AE.

5 Propuesta de Solución

Nuestro trabajo es una propuesta dirigida por modelos que identifica automáticamente elementos potencialmente redundantes apoyado en alineación de ontologías. Esta aproximación permite expresar formalmente los esquemas de datos de la organización en un repositorio central y homogéneo que favorece análisis y evaluaciones sobre los activos de datos de la organización. El sistema importa las estructuras a partir de un esquema XML o una conexión de base de datos y los consigna en un modelo XMI (XML Metadata Interchange), conforme al metamodelo Tartarus. Una vez creado el modelo, se realiza una transformación del modelo a ontologías, para inferir similitudes entre elementos utilizando un sistema de alineación de ontologías. Los resultados de estas

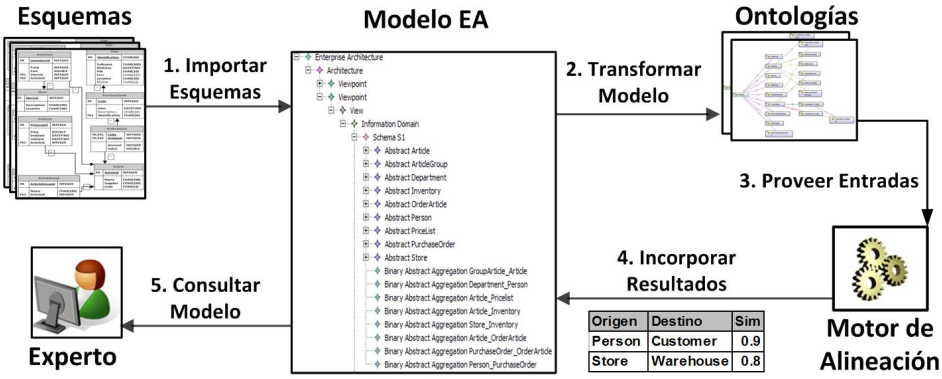

Figura 3. Diagrama general de la propuesta

inferencias son cargados de vuelta al repositorio. Nuestra propuesta se divide en cinco etapas y una vista general se muestra en la Figura 3.

5.1 Importar Esquemas

Inicialmente se requiere describir formalmente la AE, incluidos los esquemas del dominio de información. Estas definiciones pueden expresarse manualmente o importarse directamente desde un origen JDBC o XML. En ambos casos, obtenemos un modelo que contiene las entidades y sus relaciones, en términos de conceptos Tartarus. Nuestro metamodelo es una adaptación de [1], que enriquece las definiciones con las relaciones inferidas entre entidades, comentarios de tablas y comentarios de columnas, ver Figura 2B. La metaclase *Schema* representa los esquemas contenidos en la AE. Para nuestro caso, el esquema del Sistema de Inventario se convierte en la instancia *Schema:S1*. La metaclase *Attribute* está especializada en dos subclases: *SimpleAttribute*, define columnas en la base de datos o tipos primitivos en esquemas XML, tienen un tipo de dato (*INTEGER, DOUBLE, STRING*, etc). Por otro lado, *Abstract* se refiere a entidades en un modelo relacional o tipos de dato complejos en XML Schema.

Por ejemplo, la entidad *Person* se convierte en un objeto *Abstract:S1.Person* y cada uno de sus campos (*Identification, Fullname, Borndate*, etc) son objetos de tipo *SimpleAttribute* con sus respectivos tipos de dato. En *BinaryAbstractAggregation*, se definen las relaciones existentes entre cada par de elementos *Abstract*. La relación entre las entidades *Person* y *PurchaseOrder* se representa con la asociación *BinaryAbstractAggregation:Person_PurchaseOrder*. La metaclase *MatchResult* mapea correspondencias inferidas entre elementos *Abstract*, tras la ejecución del alineamiento. En un momento cero, no existen resultados de alineaciones, estos son adicionados a partir de la primera iteración. Por ejemplo, una instancia *MatchResult:S1.Person_S2.User* representa un mapeo donde, *left:S1.Person*,

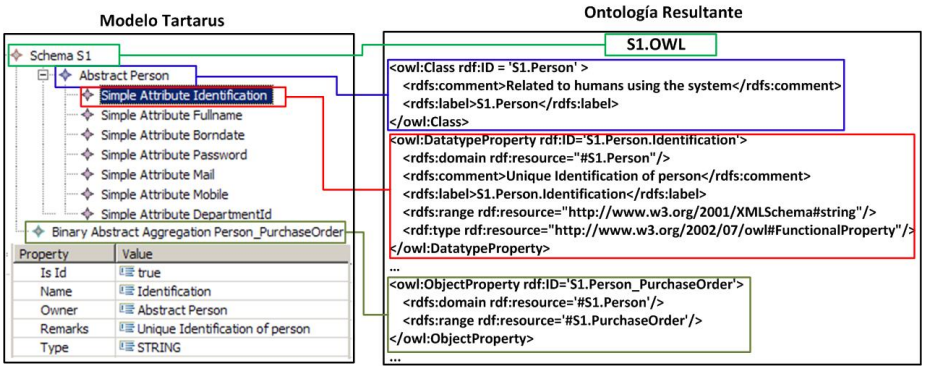

Figura 4. Transformación Tartarus-OWL

right:S2.User, *similarity:0.9* y *type:EQUIVALENCE*.

5.2 Transformar Modelo

Posteriormente, ejecutamos una transformación Tartarus-OWL para llevar todos los conceptos definidos en el modelo hacia ontologías, ver Figura 4. La decisión por OWL [2] se debió a la mayor expresividad y vocabulario que ofrece para la descripción de conocimiento, comparado con otros formatos como RDF. Se genera un archivo OWL por cada esquema definido en el modelo. Cada objeto *Abstract* es traducido a un *owl:Class*. Los elementos *SimpleAttribute* son mapeados como *owl:DatatypeProperty* de la clase OWL contenedora. El tipo de dato de dichos atributos es redefinido como dato *XMLSchema* primitivo. Sumado a esto, si el atributo está marcado como *identificador*, un elemento *owl:FunctionalProperty* es incluido. Los atributos de tipo *Abstract* son transformados a elementos *owl:ObjectProperty*, donde el dominio es la clase contenedora y el rango corresponde al atributo *Abstract*. Las instancias de *BinaryAbstractAggregation* también se convierten en *owl:ObjectProperty*, donde los *Abstract* origen y destino se homologan a dominio y rango respectivamente. Finalmente, los *remarks* del modelo se trasladan como *rdfs:comment* de clases y propiedades OWL.

5.3 Proveer Entradas

El siguiente paso consiste en proveer las ontologías de entrada al sistema de alineamiento. Para lo cual, realizamos una combinatoria en la que, de un universo de n ontologías, tomamos un subconjunto k, de dos elementos (ontología origen y destino) para ser comparados. El total de comparaciones esta dado por la fórmula de Coeficiente Binomial $C(n,k) = \frac{n!}{k!(n-k)!}$. En nuestro caso, alinear tres esquemas implica realizar tres comparaciones: $S1xS2$, $S1xS3$ y $S2xS3$.

El motor de alineación que actualmente hace parte de nuestra solución

es AgreementMaker [5]. Esta decisión fue apoyada en términos de disponibilidad, documentación, variedad de algoritmos, extensibilidad y los resultados obtenidos [4] en la última campaña OAEI [9]. Para cada pareja de ontologías/esquemas, aplicamos un conjunto de comparadores que ya vienen implementadas dentro AgreementMaker. Cada algoritmo debe ser configurado con parámetros como umbral de similitud y cardinalidad. Estas técnicas, utilizan nombres, comentarios, etiquetas, tipos de dato y estructuras de las ontologías para determinar grados de coincidencia. Con las salidas de cada comparación, realizamos un pos-procesamiento que consiste en aplicar relación de transitividad entre alineaciones de diferentes esquemas, para inferir nuevas asociaciones.

Para explicar mejor el pos-procesamiento, asumamos que aplicamos un conjunto de algoritmos $A = \{a_1, a_2, a_3\}$ sobre los esquemas $S1$, $S2$ y $S3$. Como resultado obtenemos tres alineaciones $S1xS2 = \{S1.Person \simeq S2.Customer, S1.Store \simeq S2.Warehouse\}$, $S1xS3 = \{S1.State \simeq S3.Province\}$, $S2xS3 = \{S2.Customer \simeq S3.Client, S2.Bill \simeq S3.Invoice\}$. Note que la alineación $S1xS3$ no incluyó la correspondencia $S1.Person \simeq S3.Client$, pero basándonos en los mapeos $S1.Person \simeq S2.Customer$, $S2.Customer \simeq S3.Client$ y aplicando transitividad, podemos inferir: $S1.Person \simeq S3.Client$.

5.4 Incorporar Resultados

Los resultados de cada comparación son cargados de vuelta al modelo Tartarus. Para tal fin, implementamos una herramienta usando Eclipse Modeling Framework (EMF) que recorre el modelo y las alineaciones de salida, para asignar un índice de similitud y el tipo de relación entre cada par de objetos. Por tanto la relación $\{S1.Person \simeq S3.Client, sim : 0.8, type : EQUIVALENCE\}$ es agregada al modelo como un objeto *MatchResult:S1.Person_S3.Client*.

5.5 Consultar Modelo

Ya consolidados los resultados de la tarea de alineamiento en el modelo, se desarrollaron consultas que permiten al usuario experto visualizar y navegar por las diferentes relaciones que se detectaron. Los reportes se definieron en formato HTML (HyperText Markup Language), para facilitar la navegación a través de las entidades y sus relaciones explícitas e implícitas (inferidas). La generación del código HTML se materializa con una transformación Tartarus-HTML.

[4] Entre los tres primeros en el Benchmark Track

6 Experimentación

En nuestra experimentación, continuamos con el caso de estudio trabajado en la Sección 2, ampliándolo a todos los elementos del esquema e incluyendo dos SI adicionales. Hemos desarrollado un prototipo de nuestra propuesta como un proyecto eclipse. Dicho proyecto, contiene los metamodelos, transformaciones y herramientas de importación y alineamiento. Usando este prototipo hemos implementado un experimento de dos fases: La primera, evalúa cada una de las transformaciones en términos de rendimiento y la exactitud del proceso de alineamiento. En la segunda fase, se compara la precisión de los resultados aplicando transitividad entre diferentes alineaciones. En esta Sección, esperamos evaluar los niveles de precisión alcanzados con un primer prototipo. También medimos el aporte del pos-procesamiento, a las técnicas inciales, en términos de *F-Measure* (ver Sección 4.1). La máquina sobre la cual se realizó el experimento es un laptop con procesador de doble núcleo, 2.2 GHz de 64 bits y RAM de 4GB.

6.1 Fase 1

En esta prueba inicial cargamos cuatro esquemas. Los sistemas de inventario (S1) y CRM (S2) mencionados en la Sección 2, junto con un esquema de gestión de usuarios y roles (S3) y un sistema ERP (S4). Por medio del importador JDBC-EMF, accedimos a los esquemas para poblar el modelo Tartarus. La herramienta cargó en 890 ms la estructura de la base de datos, comprendida por: 42 tablas y 151 campos distribuidos en 4 esquemas. El 76,4% de los elementos contaban con comentarios, los cuales también fueron incluidos en el modelo para posteriormente apoyar los procesos de alineación. El modelo resultante está compuesto de 243 entidades y 1.099 atributos. El siguiente paso fue la ejecución de las transformaciones Tartarus-OWL. La salida de este proceso fueron 4 ontologías, uno por cada esquema para un total de 42 *Class*, 151 *DataProperties* y 41 *ObjectProperties* en formato OWL, con sus respectivas anotaciones. El tiempo de ejecución promedio de estas transformaciones fue de 479 ms.

La alineación se realizó programáticamente en Java, invocando iterativamente algoritmos ya incorporados en el motor de AgreementMaker. Se procesaron 6 comparaciones entre las diferentes ontologías en 56.719 ms. Los algoritmos utilizados fueron los mismos seleccionados por la propuesta AgreementMaker en el Benchmark Track de la campaña OAEI 2010: *Advanced Similarity* usa una técnica de comparación de caracteres. *Parametric String* realiza una pre-procesamiento donde limpia, lematiza y tokeniza texto para implementar comparación sintáctica. *Multi Words* es una técnica basada en similitud de espacio vectorial donde conceptos *cercanos* son tomados en cuenta. *Lexical Synonym* compara sinónimos e hiperónimos. *Partial Graph* realiza compara-

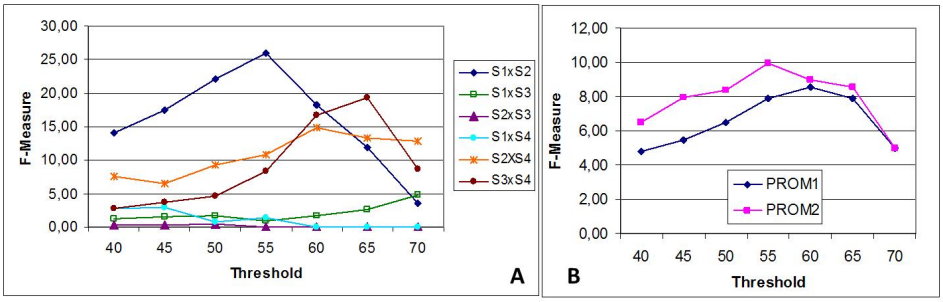

Figura 5. A) F-Measure por Umbral para cada Alineamiento . B) F-Measure Promedio por Umbral en cada Fase

ciones estructurales basadas en grafos y *Linear Weighted Combination*, un algoritmo que combina los resultados generados por los anteriores, calculando un promedio de similitud. Variamos el umbral de similitud para evaluar la precisión alcanzada en cada una de las seis comparaciones (S1xS2, S1xS3 ... S3xS4). Los resultados obtenidos fueron comparados contra *alineaciones de referencia*, las cuales son *reglas doradas* definidas por un experto. Utilizamos una alineación de referencia para evaluar la precisión de los resultados. En la Figura 5A se relacionan los niveles de exactitud (*F-Measure*) alcanzados en cada comparación por diferentes umbrales de similitud.

Pudimos verificar, como en cada comparación de pares de ontologiás se alcanzan diferentes niveles de exactitud bajo el mismo umbral de similitud. Empíricamente comprobamos, que las ontologías/esquemas que pertenecen a dominios *distantes*, es decir, ofrecían menos elementos comunes (S1xS4 y S2xS3) presentan niveles de precisión muy bajos (inferiores a 5%). Por otro lado, ontologías/esquemas de dominios *menos alejados* (S1xS2 y S3xS4), muestran mejores indicadores (26% y 20% respectivamente). En general, estos resultados preliminares, presentan niveles de calidad inferiores a los 89% obtenidos por AgreementMaker en el Benchmark Track 2010. Creemos que la principal razón, es la distancia de los dominios comparados. Mientras en el Benchmark Track se comparan ontologías de dominios afines (por ejemplo, ontologías del dominio de la Bibliografía) que tienen gran cantidad de conceptos y términos iguales o similares, nuestro trabajo compara dominios no cercanos (sistemas de inventario y CRM).

6.2 Fase 2

La segunda Fase, evalúo los resultados incorporando el pos-procesamiento para la verifición de relaciones transitivas. Se tomó el mismo conjunto de valores de umbral utilizado en la Fase 1. Calculamos para el rango de valores de umbral (40-70), el promedio de *F-Measure* de las 6 comparaciones. La Figura

5B, contrasta los promedios obtenidos en la Fase 1 (*PROM1*) y la Fase 2 (*PROM2*).

Logramos incrementar el promedio de exactitud del proceso de alineamiento, para cada uno de los niveles de umbral de similitud evaluados. El aumento mas significativo se dió con un umbral del 55% donde obtuvimos un 2% mas que lo obtenido en la Fase 1. También, evidenciamos que los mejores resultados fueron obtenidos con umbrales cercanos al 55%. Finalmente, los 2.098 mapeos obtenidos con el mejor indicador de exactitud, fueron incorporados al modelo Tartarus. Un total de 2.102 entidades, 8.408 atributos y 4.204 asociaciones fueron adicionadas al modelo para almacenar estas inferencias en 3.235 ms. De esta manera, las relaciones inferidas trazaron nuevos vínculos entre elementos de la AE, que aumentan su completitud.

7 Trabajo Relacionado

Protégé [15] es un entorno para diseñar ontologías ampliamente utilizado, y gracias a su arquitectura extensible, permite adicionar plug-ins para usos específicos. Entre ellos, se encuentran algunos que permiten importar esquemas relacionales y XML a ontologías.

En [30] se trata la detección de redundancias de datos; pero aplicado a estructuras XML y basado solo en su contenido. Encontrar similitudes en procesos de negocio, ha sido un tema abordado por otros autores. En [6] se propone la aplicación de alineaciones basadas en grafos y léxicos para encontrar actividades similares en modelos de procesos de negocio. Por otro lado, una propuesta para expresar procesos de negocios con Redes de Petri sobre ontologías, para realizar alineaciones semánticas se expone en [3].

Numerosas investigaciones han abordado el alineamiento de esquemas [25][27] e implementado herramientas para automatizarlo [20][7][8][28][14][5]. Estas aproximaciones alinean ontologías y exportan los mapeos resultantes a formatos propios, hojas de cálculo, archivos planos, OWL o RDF. RiMoM [17] es un framework multi-estrategia de alineamiento de ontologías que automáticamente combina diferentes técnicas y estuvo entre las tres mejores propuestas del Benchmark Track de la OAEI 2010. Este sistema permite configurar conexiones de base de datos para comparar esquemas.

Nuestra propuesta, a diferencia de estos trabajos, integra, centraliza y enriquece el alineamiento de esquemas, dado que nuestro metamodelo define el dominio de información y sus entidades, no de manera aislada, sino integrado con los demás elementos que componen la AE.

8 Conclusiones

Hemos presentado una propuesta para detectar semi-automáticamente elementos redundantes en esquemas de datos, en el marco de una AE. Extendimos un Tartarus para incluir relaciones explícitas e implícitas entre entidades de datos. Explotamos un sistema para generar alineaciones ontológicas de estas entidades mediante diferentes técnicas. Incorporamos los resultados de la alineación a la AE, para facilitar procesos de anális en la arquitectura de información. Contrastamos las diferencias, en cuanto a niveles de exactitud, al comparar ontologías de dominios cercanos y dominios distantes. Realizamos pruebas que muestran, como aplicando transitividad, mejoramos la exactitud de un proceso de alineación.

Dados los índices de exactitud obtenidos en nuestra experimentación, no podemos omitir la interacción con un experto que verifique y confirme los mapeos a fin de mejorar la confiabilidad de los mismos. Hemos demostrado que es posible apoyar el proceso de identificar redundancias potenciales, de forma semi-automática, acortando los tiempos de comparación y consolidando los mapeos resultantes. Actualmente estamos trabajando en la inclusión de otros dominios de la AE (negocio, aplicaciones, servicios, etc) en los procesos de alineación, tanto para mejorar la exactitud de los resultados, como para identificar posibles redundancias en dichos dominios.

Bibliografía

[1] Atzeni, P., P. Cappellari, R. Torlone, P. A. Bernstein and G. Giorgio, *Model-independent schema translation*, The VLDB Journal **17** (2008), pp. 1347–1370.

[2] Bechhofer, S., F. van Harmelen, J. Hendler, I. Horrocks, P. F. P.-S. Deborah McGuinness and L. A. Stein, *Owl web ontology language reference*, W3C Working Draft (2004).

[3] Brockmans, S., M. Ehrig, A. Koschmider, A. Oberweis and R. Studer, *Semantic alignment of business processes*, Proceedings of the Eighth International Conference on Enterprise Information Systems (ICEIS 2006) (2006), pp. 191–196.

[4] Cook, M. A., "Building enterprise information architectures: reengineering information systems," Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.

[5] Cruz, I. F., F. P. Antonelli and C. Stroe, *Agreementmaker: efficient matching for large real-world schemas and ontologies*, Proc. VLDB Endow. **2** (2009), pp. 1586–1589.

[6] Dijkman, R., M. Dumas, L. Garcia-Banuelos and R. Kaarik, *Aligning business process models*, Enterprise Distributed Object Computing Conference, IEEE International **0** (2009), pp. 45–53.

[7] Ehrig, M. and S. Staab, *Qom quick ontology mapping*, in: S. McIlraith, D. Plexousakis and F. van Harmelen, editors, *The Semantic Web ISWC 2004*, Lecture Notes in Computer Science **3298**, Springer Berlin / Heidelberg, 2004 pp. 683–697.

[8] Ehrig, M., S. Staab and Y. Sure, *Bootstrapping ontology alignment methods with apfel*, in: Y. Gil, E. Motta, V. Benjamins and M. Musen, editors, *The Semantic Web ISWC 2005*, Lecture Notes in Computer Science **3729**, Springer Berlin / Heidelberg, 2005 pp. 186–200.

[9] Euzenat, J., A. Ferrara, C. Meilicke, A. Nikolov, J. Pane, F. Scharffe, P. Shvaiko, H. Stuckenschmidt, O. Svab-Zamazal, V. Svatek and C. Trojahn, *First results of the ontology alignment evaluation initiative 2010*, Workshop on Ontology Matching **10** (2010), pp. 334–350.

[10] Euzenat, J. and P. Shvaiko, "Ontology Matching," Springer Berlin / Heidelberg, 2007.

[11] Gal, A. and P. Shvaiko, *Advances in ontology matching*, in: T. Dillon, E. Chang, R. Meersman and K. Sycara, editors, *Advances in Web Semantics I*, Lecture Notes in Computer Science **4891**, Springer Berlin / Heidelberg, 2009 pp. 176–198.

[12] Gaŝevic, D., D. Djuric and V. Devedzic, "Model Driven Engineering and Ontology Development," Springer Berlin / Heidelberg, 2009.

[13] Group, D. A. F. W., *Dod architecture framework version 1.0* (2005).
URL http://www.defenselink.mil/nii/doc/DoDAF_vI_Volume_II_pdf

[14] Jean-Mary, Y. R., E. P. Shironoshita and M. R. Kabuka, *Ontology matching with semantic verification*, Web Semantics: Science, Services and Agents on the World Wide Web **7** (2009), pp. 235 – 251.

[15] Knublauch, H., R. Fergerson, N. Noy and M. Musen, *The protégé owl plugin: An open development environment for semantic web applications*, in: S. McIlraith, D. Plexousakis and F. van Harmelen, editors, *The Semantic Web ISWC 2004*, Lecture Notes in Computer Science **3298**, Springer Berlin / Heidelberg, 2004 pp. 229–243.

[16] Küngas, P. and M. Dumas, *Redundancy detection in service-oriented systems*, in: *Proceedings of the 19th international conference on World wide web* (2010), pp. 581–590.

[17] Li, J., J. Tang, Y. Li and Q. Luo, *Rimom: A dynamic multistrategy ontology alignment framework*, IEEE Transactions on Knowledge and Data Engineering **21** (2009), pp. 1218–1232.

[18] Liu, H., X. Wang and Q. Quan, *Research on the enterprise' model of information lifecycle management based on enterprise architecture*, , **3**, 2009, pp. 165 –169.

[19] Madhavan, J., P. A. Bernstein, A. Doan and A. Halevy, *Corpus-based schema matching* **0** (2005), pp. 57–68.

[20] Madhavan, J., P. A. Bernstein and E. Rahm, *Generic schema matching with cupid*, in: *Proceedings of the 27th International Conference on Very Large Data Bases* (2001), pp. 49–58.

[21] Moody, D. L. and G. G. Shanks, *Improving the quality of data models: empirical validation of a quality management framework*, Inf. Syst. **28** (2003), pp. 619–650.

[22] MoosasGroup, *Tartarus enterprise architecture meta-model* (2011).
URL http://moosas.uniandes.edu.co/doku.php?id=tartarus

[23] Noy, N. F. and D. L. Mcguiness, *Ontology Development 101: A Guide to Creating Your First Ontology*, Technical report, Stanford Knowledge Systems Laboratory (2001).

[24] Pereira, C. M. and P. Sousa, *A method to define an enterprise architecture using the zachman framework*, in: *Proceedings of the 2004 ACM symposium on Applied computing*, SAC '04 (2004), pp. 1366–1371.

[25] Rahm, E. and P. A. Bernstein, *A survey of approaches to automatic schema matching*, The VLDB Journal **10** (2001), pp. 334–350.

[26] Rodriguez, M. E., F. Murcia and D. E. Correal, *Tartarus una estrategia para construir y expresar arquitecturas empresariales*, Revista Avances en Sistemas e Informática **8** (2011), pp. 61–70.

[27] Shvaiko, P. and J. Euzenat, *A survey of schema-based matching approaches*, in: S. Spaccapietra, editor, *Journal on Data Semantics IV*, Lecture Notes in Computer Science **3730**, Springer Berlin / Heidelberg, 2005 pp. 146–171.

[28] Tang, J., J. Li, B. Liang, X. Huang, Y. Li and K. Wang, *Using bayesian decision for ontology mapping*, Web Semantics: Science, Services and Agents on the World Wide Web **4** (2006), pp. 243–262.

[29] TheOpenGroup, "TOGAF Version 9," Van Haren Publishing, 2008.

[30] Yu, C. and H. V. Jagadish, *Efficient discovery of xml data redundancies*, in: *Proceedings of the 32nd international conference on Very large data bases* (2006), pp. 103–114.

[31] Zachman, J. A., *A framework for information systems architecture*, IBM Syst. J. **26** (1987), pp. 276–292.