iTrace: un framework para soportar el análisis de información de trazabilidad en proyectos de Desarrollo Software Dirigidos por Modelos

Iván Santiago, Juan M. Vara, Valeria de Castro, Esperanza Marcos,

Grupo de Investigación Kybele, Departamento de Lenguajes y Sistemas Informáticos II, Universidad Rey Juan Carlos, Avda. Tulipán S/N, 28933, Móstoles, Madrid, España

{ivan.santiago, juanmanuel.vara, valeria.decastro, esperanza.marcos}@urjc.es

Resumen. El papel clave de los modelos en cualquier proceso de Desarrollo de Software Dirigido por Modelos (DSDM) proporciona un nuevo escenario para manejar la trazabilidad. Hasta ahora, existen una serie de propuestas dedicadas al almacenamiento, visualización, generación semi-automática y gestión de operaciones CRUD con enlaces de traza. No obstante, existe una falta de propuestas que se centren en el análisis de la información proporcionada por tales trazas. Además, las propuestas que llevan a cabo algún tipo de análisis de la información de trazabilidad no tienen en cuenta que esta información es consumida por diferentes tipos de actores, cada uno con sus propias necesidades. Para abordar estas cuestiones en este trabajo se introduce iTrace, un framework para la gestión y el análisis de la información de trazabilidad en proyectos de DSDM. Nuestra propuesta busca mejorar el uso que se hace de la información de trazabilidad disponible en proyectos de DSDM mediante dos tipos diferentes de análisis: análisis orientado a modelos, para modeladores, desarrolladores y el resto de perfiles operativos y análisis orientado a datos, para jefes de proyecto, analistas de negocio y usuarios finales en general.

Palabras Claves: Desarrollo Software Dirigido por Modelos, Análisis de Información de Trazabilidad, Análisis Orientado a Modelos, Análisis Orientado a Datos, Modelos de Traza

1 Introducción

La trazabilidad [1] ha sido siempre un tema relevante en la Ingeniería del Software [2]. Mantener enlaces entre los requisitos, los artefactos de análisis y diseño, el código o los casos de uso, ha resultado útil como una forma de llevar a cabo las pruebas de regresión, la validación de requisitos, etc. Del mismo modo, una gestión adecuada de la información de trazabilidad es clave para controlar la evolución de los diferentes componentes del sistema a lo largo del ciclo de vida del desarrollo software [3].

Desafortunadamente, el mantenimiento de estos enlaces es un proceso tedioso, lento y propenso a cometer errores si no se proporcionan herramientas que automaticen total o parcialmente la tarea [4]. Como consecuencia, la información de trazabilidad se convierte en obsoleta muy rápidamente durante el desarrollo del software y, a veces, se omite completamente. La llegada de la Ingeniería Dirigida por Modelos (IDM) [5] puede cambiar drásticamente esta situación. El impacto de la IDM se ha traducido en la aparición de una serie de propuestas metodológicas para el Desarrollo Software Dirigido por Modelos (DSDM) [6], donde el rol clave que juegan los modelos puede influir positivamente en la gestión de la información de trazabilidad, ya que en un proceso de DSDM las trazas entre artefactos software que tienen que ser mantenidas son, principalmente, enlaces entre los elementos de los diferentes modelos manejados a lo largo del proceso.

En este sentido, en trabajos previos [7] hemos llevado a cabo una revisión sistemática de la literatura para evaluar el estado del arte sobre la gestión de la trazabilidad en el contexto del DSDM. Una de las principales conclusiones de dicha revisión fue que las operaciones más comúnmente tratadas por las propuestas existentes son el almacenamiento, la visualización y las operaciones de creación, recuperación modificación y borrado (CRUD, *Create*, *Read*, *Update* y *Delete*) de enlaces de traza. Por el contrario, las operaciones menos comúnmente tratadas son el intercambio y el análisis de la información de trazabilidad.

Además, las pocas propuestas que realizan o proponen algún tipo de análisis, no consideran que la información de trazabilidad es "consumida" por diferentes tipos de actores, cada uno con sus propias necesidades y objetivos respecto a la información de trazabilidad [8].

Para hacer frente a este problema, en este trabajo se introduce el primer prototipo de iTrace: un framework para soportar la gestión y el análisis de información de trazabilidad en proyectos de DSDM. Para dar respuesta a las necesidades de los diferentes actores respecto a la información de trazabilidad, iTrace soporta dos tipos de análisis. Por una parte, los modeladores, desarrolladores y el resto de perfiles operativos necesitan de información de bajo nivel. Para hacer frente a sus necesidades, iTrace soporta el Análisis Orientado a Modelos, cuyos resultados se proporcionan en forma de modelos de traza que pueden ser posteriormente procesados mediante técnicas de IDM [9]. Por otra parte, las necesidades de los analistas de negocio, jefes de proyecto y usuarios finales van más en la línea de disponer de datos que apoyen los procesos de toma de decisiones y que permitan elicitar conocimiento a partir de la información de trazabilidad (de bajo nivel) presente en cualquier proyecto de DSDM. Para hacer frente a estas necesidades, iTrace soporta el Análisis Orientado a Datos que produce información de alto nivel en forma de datos agregados.

El resto del trabajo se estructura de la siguiente forma. La sección 2 revisa los trabajos relacionados en el área. La sección 3 presenta la propuesta iTrace, detallando el proceso y los componentes técnicos que la conforman. La sección 4 ilustra la propuesta mediante un caso de estudio, mientras que la sección 5 discute las limitaciones actuales e identifica las principales líneas de trabajo futuro para solventarlas. Finalmente, la sección 6 resume las principales conclusiones derivadas de este trabajo.

2 Trabajos relacionados

Antes de comenzar la revisión de los trabajos relacionados, nos gustaría concretar algunos de los términos relacionados que se utilizan a lo largo de este trabajo. Definimos una *relación de trazabilidad* como una correspondencia entre dos o más tipos de elementos (por ejemplo, entre clases). Por otro lado, nos referimos a las instancias de estas relaciones como *enlaces de traza* (o simplemente trazas). Finalmente, la *información de trazabilidad* es generalmente obtenida a partir de uno o más enlaces de traza; es decir, los enlaces de traza son la materia prima para la construcción de información de trazabilidad.

Existen varios trabajos relacionados con la gestión de la trazabilidad en el contexto del DSDM. La mayoría de estos trabajos consideran el almacenamiento, visualización, generación semi-automática y la realización de operaciones CRUD de enlaces de traza, pero solo unos pocos de esos trabajos abordan el análisis de la información de trazabilidad que proporcionan dichas trazas. Estos trabajos pueden ser clasificados en tres grandes categorías de acuerdo con su objetivo principal: 1) análisis del impacto [10–12]; 2) generación de informes de trazabilidad [13, 14] e 3) identificación y clasificación de trazas [15].

Respecto a los trabajos centrados en el análisis del impacto, los trabajos de Anquetil et al. [10] y Walderhaug et al. [12] presentan propuestas para la extracción de información de un repositorio de enlaces de traza con el objetivo de identificar todos los artefactos potencialmente afectados por un cambio. Además, el primer trabajo acompaña la propuesta de una herramienta de soporte. Por otra parte, Olsen y Oldevik [11] presentan un prototipo donde los artefactos enlazados son mostrados al seleccionar los elementos del modelo origen. Estas propuestas ayudan en la identificación de los elementos que pueden verse afectados por un determinado cambio, ofreciendo el mismo tipo de información independientemente del actor que solicita dicha información.

A continuación consideramos dos herramientas diferentes centradas en la generación de informes de trazabilidad: TraceTool and Safety Requirements Trace Tool. La primera herramienta, presentada por Valderas y Pelechano en [14], genera informes de trazabilidad en formato HTML que describen cómo los elementos de un modelo navegacional se derivan de un modelo de requisitos. La segunda herramienta, presentada en [13] por Sánchez et al., consume modelos de traza para generar informes de trazabilidad en formato HTML, en el contexto del DSDM para robots de servicios tele-operados. Nótese que ambas herramientas están centradas en la trazabilidad de requisitos, es decir, no consideran la trazabilidad de más bajo nivel, como las trazas que pueden ser derivadas de una transformación de modelos de bajo nivel o una transformación modelo a texto.

Finalmente, en [15] Paige et al. presentan *Traceability Elicitation and Analysis Process* (TEAP), una propuesta para la identificación y clasificación de relaciones de trazabilidad. Clasificar los enlaces de trazabilidad ayuda a entenderlos y manejarlos. No obstante, a pesar de tratarse de una propuesta muy útil para los miembros de la comunidad de la IDM, puede ser difícil de comprender para un analista de negocio o un usuario final.

iTrace proporciona una serie de contribuciones respecto al estado del arte actual: en primer lugar, la propuesta se centra en los artefactos software de un proyecto de DSDM que implícitamente definen las relaciones de trazabilidad que deben ser monitorizadas (además de los requisitos, ampliamente abordados en la literatura): no solo transformaciones de modelo a modelo, sino también modelos de anotación y modelos de *weaving* en general [9]. También soporta dos tipos de análisis para dos grupos principales de actores: Análisis Orientado a Modelos y Análisis Orientado a Datos. Además, podríamos decir que todos los trabajos revisados, salvo el de Walderhaug et al. [12] proporcionan un análisis ad-hoc. Es decir, el proceso de análisis propuesto encaja en sus propios proyectos de DSDM, mientras que iTrace pretende proporcionar una propuesta genérica que se pueda aplicar a cualquier proyecto existente de DSDM. Por último, otra de las características relevantes en relación a las propuestas existentes es el hecho de que iTrace soporta el análisis multidimensional de los enlaces de traza.

3 iTrace: un framework para soportar análisis de información de trazabilidad

Esta sección presenta la propuesta para soportar el análisis de información de trazabilidad que se introduce en este trabajo, presentando para ello los principales componentes del entorno iTrace y el proceso de generación y análisis de trazas que soporta.

3.1 Proceso soportado iTrace

El punto de partida del proceso de análisis soportado por iTrace es un proyecto de DSDM existente. Téngase en cuenta que la propuesta pretende ser aplicable a cualquier tipo de proyecto. Por lo tanto, consideramos como punto de partida un proyecto genérico, es decir, un proyecto que incluye modelos a diferentes niveles de abstracción (incluidos modelos de *weaving*) además de un conjunto de transformaciones de modelos que los conectan (transformaciones modelo a modelo y modelo a texto), donde los modelos iniciales (modelos con nivel de abstracción más alto) se transforman posteriormente en modelos de nivel inferior, hasta que su nivel de abstracción permite utilizarlos para generar código.

La parte izquierda de la Fig. 1, muestra una versión simplificada de un proyecto de este tipo, compuesto por tres modelos llamados *Ma*, *Mb* y *Mc* y dos transformaciones de modelo a modelo (*Ma2Mb* y *Mb2Mc*).

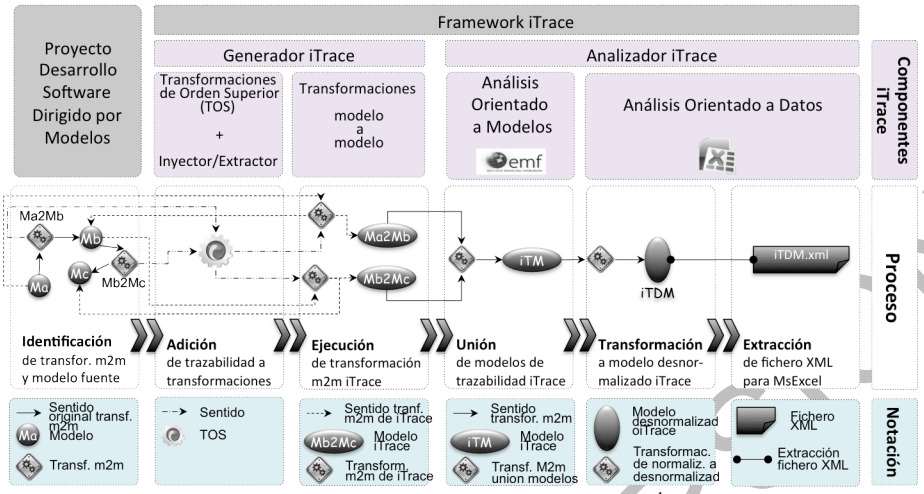

Fig. 1. Visión general del entorno iTrace[1]

El resto de la Fig. 1 proporciona una visión tabular del framework iTrace: la fila superior muestra los **Componentes** involucrados en cada paso del proceso; la siguiente fila muestra los artefactos (modelos, transformaciones, etc.) usados y producidos en cada paso del **Proceso**; finalmente, la última fila muestra la **Notación** utilizada para comprender los elementos gráficos que se utilizan en cada celda.

Si nos centramos en el proceso, la fase de **Identificación** representa el inicio del mismo, donde un proyecto existente de DSDM compuesto por modelos y transformaciones es seleccionado como entrada. Durante la fase de **Adición**, cada transformación del proyecto es enriquecida mediante una Transformación de Orden Superior (TOS) [16] siguiendo la idea esbozada por Jouault en [17]. A continuación, las transformaciones enriquecidas son ejecutadas durante la fase de **Ejecución** con el fin de generar modelos de traza además de los correspondientes modelos destino. Estos modelos de traza son consolidados en un único modelo durante la fase de **Unión**. Este modelo agregado de trazas es el que sirve de entrada para llevar a cabo el Análisis Orientado a Modelos soportado por iTrace.

Podemos decir que este modelo agregado de trazas es un modelo normalizado. Sin embargo, a la hora de ejecutar las consultas que realizan el Análisis Orientado a Datos soportado por iTrace hemos encontrado más conveniente utilizar un modelo de trazas desnormalizado. Adoptamos esta idea del área de las bases de datos, donde los modelos de datos normalizados son más convenientes para adiciones, modificaciones y borrados, mientras que se utilizan modelos de datos desnormalizados para mejorar el rendimiento de las consultas en sistemas de consulta intensiva, como los almacenes de datos (datawarehouses) [18]. En cierto sentido, el análisis orientado a datos de información de trazabilidad puede ser visto como el análisis de los datos históricos recopilados durante el desarrollo del proyecto, principalmente en forma de modelos de traza. Por lo tanto, durante la fase de **Transformación**, una transformación modelo a

[1] Todas las imágenes se encuentran disponibles a tamaño completo para su mejor visualización en: http://www.kybele.etsii.urjc.es/JISBD/iTrace/

modelo consume el modelo de trazas normalizado generado en la fase de unión (modelo normalizado iTrace) para producir un modelo de trazas desnormalizado (modelo desnormalizado iTrace).

Por último, durante la fase de **Extracción**, el modelo desnormalizado de trazas es serializado en un fichero XML que después será importado a una hoja de cálculo de Excel con el fin de poblar la tabla dinámica que proporciona la base sobre la que se realiza el Análisis Orientado a Datos en la versión actual del prototipo. Nótese que el objetivo de este trabajo era fundamentalmente evaluar la viabilidad de nuestra propuesta. En trabajos futuros evaluaremos la posibilidad de utilizar sistemas más sofisticados para soportar el análisis orientado a datos.

Conviene destacar que hasta ahora se ha presentado un proceso independiente de la tecnología para el análisis de trazabilidad, es decir, el proceso descrito podría ser implementado sobre cualquier marco de metamodelado que ofrezca soporte para el desarrollo de transformaciones de modelos (tanto de modelo a modelo, como de modelo a texto).

A continuación, profundizamos en los detalles de la propuesta mostrando, a modo de prueba de concepto, la implementación de los diferentes artefactos software que componen iTrace utilizando Eclipse Modeling Framework (EMF) [19] como base tecnológica.

3.2 Modelos iTrace

A lo largo del proceso soportado por iTrace se generan dos tipos de modelos de traza: modelos de traza normalizados y modelos de traza desnormalizados. A continuación se presentan brevemente los metamodelos correspondientes para ilustrar la sintaxis abstracta de estos modelos. En cuanto a la sintaxis concreta, hasta ahora utilizamos una versión mejorada de los editores en forma de árbol generados por EMF.

Metamodelo iTrace

El metamodelo de iTrace, que se muestra en la Fig. 2, soporta el modelado de la información de trazabilidad de bajo nivel obtenida a partir de transformaciones de modelos (modelo a modelo y modelo a texto) y modelos *weaving*.

Un `iTraceModel` (clase raíz) contiene `Artifacts` (software) y/o `TraceLinks`. El primero representa los bloques de construcción del proyecto de DSDM, es decir, modelos y código fuente, mientras que el segundo representa las trazas entre ellos. En realidad, estas trazas conectan sus componentes, representados mediante objetos `TraceLinkElement` en el caso de los modelos y `Blocks` en el caso del código fuente. Cada objeto `TraceLinkElement` tiene una referencia `EObject` para apuntar a un elemento del modelo en particular.

Por su parte, cada traza puede ser un `TransformationLink`, derivada de una transformación de modelos, o un `AnnotationLink`, derivada de un modelo de *weaving*. A su vez, la clase `TransformationLink` se especializa en las clases `M2MLink` y `M2TLink`. Mientras que un objeto `M2MLink` relaciona dos o más elementos del modelo (al menos un `TraceLinkSourceElement` y un `TraceLinkTargetElement`), un objeto `M2TLink` relaciona uno o más elementos del modelo con algún bloque de código fuente (`M2TLink.codeTarget.blockCode`).

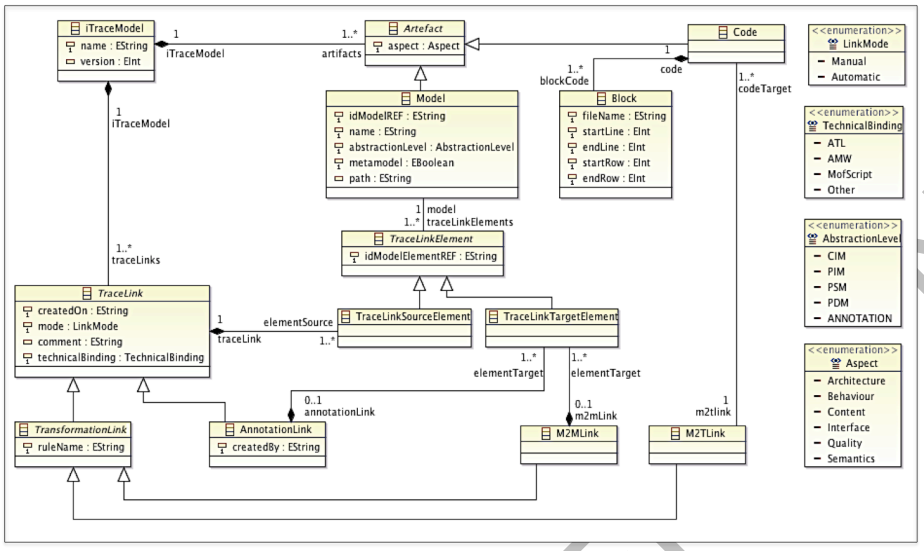

Fig. 2. Metamodelo iTrace

Metamodelo desnormalizado iTrace

El metamodelo desnormalizado de iTrace está diseñado para soportar un paso intermedio en el proceso soportado por iTrace. En particular, la información de trazabilidad recogida en los diferentes modelos iTrace (en la siguiente sección se mostrará como se generan) son consumidos por una transformación ATL que produce información "agregada" en forma de un modelo desnormalizado.

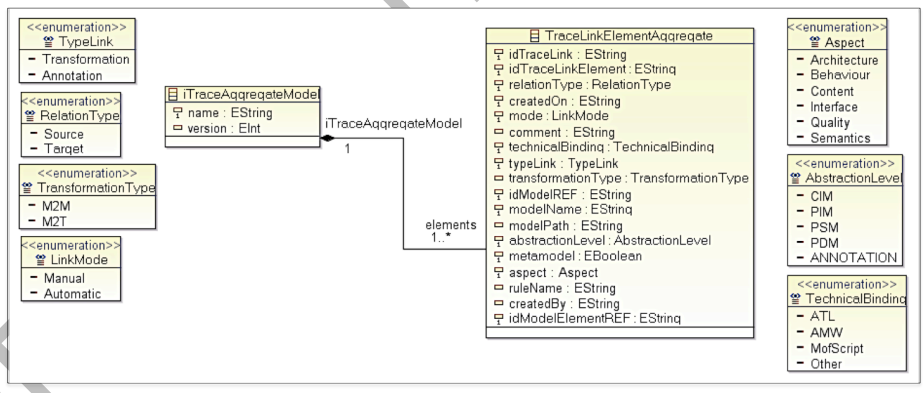

Fig. 3. Metamodelo desnormalizado iTrace

El metamodelo correspondiente, que se muestra en la Fig. 3, contiene dos clases, además de siete tipos de datos enumerados. De esta manera, cada `iTraceDenormalizedModel` se compone de objetos `DenormalizedTraceLinkElements` que poseen un conjunto de atributos que toman su valor de los diferentes tipos enumerados, facilitando las consultas que soportan el Análisis Orientado a Datos.

3.3 Generación de trazas en iTrace

Como se muestra en la Fig. 1, podemos distinguir dos componentes principales en la arquitectura de iTrace: el Generador iTrace diseñado para soportar la extracción de enlaces de traza de un proyecto existente de DSDM, y el Analizador iTrace que procesa la información proporcionada por tales trazas. A continuación presentamos el primer componente basado en el uso de Transformaciones de Orden Superior.

Tenga en cuenta que la versión actual de iTrace soporta la extracción de enlaces de traza a partir de transformaciones modelo a modelo (m2m) escritas en ATL [20], transformaciones modelo a texto (m2t) escritas en lenguaje MOFScript [21] y modelos de *weaving* elaborados con la herramienta Atlas Model Weaver (AMW) [22]. Sin embargo, vale la pena señalar que extender la propuesta para soportar cualquier otro lenguaje de transformación de modelos es técnicamente factible. De hecho, sería casi inmediato si se tratase de un lenguaje basado en un metamodelo de forma que soportase el modelado de las transformaciones incluidas en el proyecto de DSDM en cuestión.

iTrace se apoya en una Transformación de Orden Superior (TOS) [16] para enriquecer las transformaciones m2m existentes para que sean capaces de producir, no solo los modelos destino correspondientes, sino también modelos de traza. Esta idea fue propuesta por Jouault en [17], donde presentaba un prototipo inicial que permitía añadir mecanismos para la generación de trazas a transformaciones ATL existentes. En este caso, el proceso de enriquecimiento de transformaciones m2m soportado por iTrace es un poco más complejo que el que se presentaba en [17], debido al incremento de la complejidad de los metamodelos de iTrace.

La Fig. 4 muestra gráficamente el proceso: en primer lugar, el inyector/extractor TCS [23] para ficheros ATL incluido en la plataforma Atlas Model Management Architecture (AMMA) [24, 25] produce un modelo de transformación a partir del código ATL de una determinada transformación (a); este modelo de transformación es enriquecido mediante una TOS con construcciones para la generación de trazas (b) y finalmente el modelo de transformación resultante es serializado en otra transformación ATL (c). Como se mencionó anteriormente, la ejecución de esta transformación enriquecida producirá, no solo los correspondiente modelos de destino, sino también uno o varios modelos de trazas.

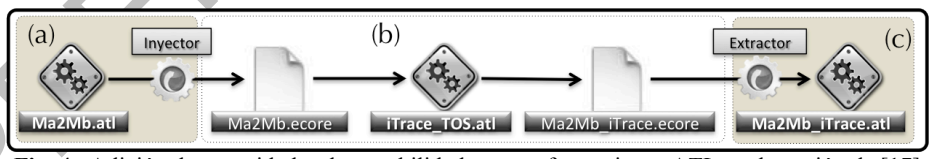

Fig. 4. Adición de capacidades de trazabilidad en transformaciones ATL – adaptación de [17]

3.4 Análisis de trazas en iTrace

Después de mostrar el proceso de generación de enlaces de traza soportado por iTrace, introducimos el proceso de análisis de dichas trazas. Para ello, en esta sección se describen los dos tipos de análisis soportados por iTrace: Análisis Orientado a Modelos y Análisis Orientado a Datos.

Análisis Orientado a Modelos

El Análisis Orientado a Modelos tiene por objetivo ofrecer información que satisfaga las necesidades de los actores involucrados en un proyecto de DSDM con un perfil operacional. Hablamos en general de modeladores, desarrolladores de transformaciones, etc. Los artefactos que suelen manejar este tipo de actores son modelos, por lo tanto el resultado de este tipo de análisis habrá de tomar la forma de modelos. Por ello, la idea subyacente es filtrar los elementos de los modelos de traza existentes atendiendo a ciertos criterios, con el fin de producir nuevos modelos de traza.

Actualmente, existen diferentes formas de consultar el contenido de uno o varios modelos. Las alternativas van desde el uso de GPLs como Java, a aproximaciones más específicas como el uso del lenguaje OCL [26] o herramientas ad-hoc, como EMF Query [27], EMF Query 2 [28] o VirtualEMF [29]. Estos últimos encapsulan gran parte de la complejidad asociada con la navegación de un modelo y podríamos decir que proporcionan las ventajas tradicionales de seguir una aproximación específica de domino en lugar de una aproximación de propósito general [30].

En particular, la implementación actual de iTrace utiliza EMFQuery porque era la iniciativa más madura entre las herramientas evaluadas. Sin embargo, presenta algunas limitaciones en cuanto a la navegación de modelos, ya que solo permite la consulta de objetos directamente anidados en el objeto raíz del modelo (metaclase `iTraceModel` en la Fig. 2). Por lo tanto, la versión actual de iTrace soporta únicamente consultas de recuperación de enlaces de traza, que pueden filtrase de acuerdo a:

- Artefacto software del que se derivan: transformación ATL, transformación MOFScript, modelo AMW, Others.
- Modo de generación: automático, manual.
- Cardinalidad del enlace: 1:1, 1:N, N:1, N:M.

Análisis Orientado a Datos

El objetivo del Análisis Orientado a Datos es obtener información que ofrezca soporte a la toma de decisiones, creando conocimiento a partir de las relaciones de trazabilidad existentes en el proyecto de DSDM bajo análisis. Frente a los modelos de traza (filtrados) producidos por el Análisis Orientado a Modelos, los resultados del Análisis Orientado a Datos son datos textuales.

En general, las organizaciones de desarrollo software tratan de identificar nuevas oportunidades y de mejorar su productividad con el fin de tener una ventaja competitiva. Para ello, es clave disponer de información analítica y estratégica, que se obtiene a partir de los datos que a diario genera y almacena la organización [31]. Este proceso de extracción y análisis de información se lleva a cabo mediante sistemas de inteligencia de negocio [32] que operan sobre soluciones basadas en almacenes de datos [33]. La idea subyacente es utilizar datos operacionales para producir información relevante a nivel táctico o incluso estratégico. En nuestra opinión, sería interesante trasladar este enfoque al uso de la información de trazabilidad que se puede generar en un proyecto de DSDM.

Para ilustrar esta idea, la siguiente sección introduce mínimamente un caso de estudio que describe un escenario básico de Análisis Orientado a Datos de información de trazabilidad.

4 Caso de Estudio

En esta sección se muestra la aplicación de la propuesta en un proyecto de DSDM existente (nótese que por motivos de espacio nos limitamos a mostrar una pequeña parte del caso de estudio).

Para ello, nos apoyamos en M2DAT-DB [34], un framework para Desarrollo Dirigido por Modelos de esquemas de bases de datos modernos que soportan la totalidad del ciclo de desarrollo, desde el nivel PIM hasta el código. En particular, M2DAT-DB incluye soporte para la generación de esquemas ORDB para Oracle y el estándar SQL:2003, así como esquemas XML a partir del modelo conceptual de datos representado con un diagrama de clases UML. En este trabajo nos centramos en tres transformaciones de modelos diferentes: la transformación UML2SQL2003 produce un modelo estándar ORDB (Modelo Específico de Plataforma, PSM) a partir de un diagrama de clases UML (Modelo Independiente de Plataforma, PIM), mientras que la transformación SQL20032ORDB genera un modelo ORDB para Oracle (Modelo Dependiente de Plataforma, PDM) a partir del modelo para el estándar. Por último, la transformación ORDB2SQL2003 implementa la transformación inversa. Además, aparte de los correspondientes modelos origen, cada transformación es capaz de consumir un modelo de anotación para introducir algunas decisiones de diseño en el proceso de transformación [35].

En el proyecto que se analiza con iTrace, estas transformaciones son ejecutadas utilizando el caso de estudio Online Movie Database (OMDB) de Feuerlicht et al. [36]. El uso de un caso de estudio "externo" evita la utilización de modelos ad-hoc que pudieran encajar mejor con nuestros propósitos.

Siguiendo el proceso descrito en la Sección 3.1 se genera un modelo agregado de trazas de trazas que es transformado en un modelo desnormalizado que es la entrada del proceso de Análisis Orientado a Datos. Dicho modelo se serializa en un fichero XML que se utiliza para poblar la tabla dinámica de MS Excel que soporta el Análisis Orientado a Datos.

La Fig. 5 muestra la información que permite analizar el rol y el nivel de abstracción de los modelos involucrados en las diferentes transformaciones. La parte izquierda de la figura muestra la información de forma tabular (1):

- En la parte superior se muestran los criterios de alto nivel que pueden ser utilizados. De esta forma, la información generada puede ser filtrada de acuerdo a la versión del modelo de trazas (1.0 en este caso); la fecha de creación (20 de Diciembre de 2011); el modo de creación (automático); o el aspecto modelado por los modelos considerados (contenido).
- El eje vertical corresponde al nivel de abstracción de los elementos del modelo trazado (PDM, PIM o PSM) y pueden ser refinados de acuerdo al rol de los elementos del modelo en el enlace de traza (origen o destino).
- El eje horizontal considera las diferentes transformaciones involucradas en el proyecto. Se puede ofrecer mayor nivel de detalle incluyendo los modelos concretos implicados en dichas transformaciones.

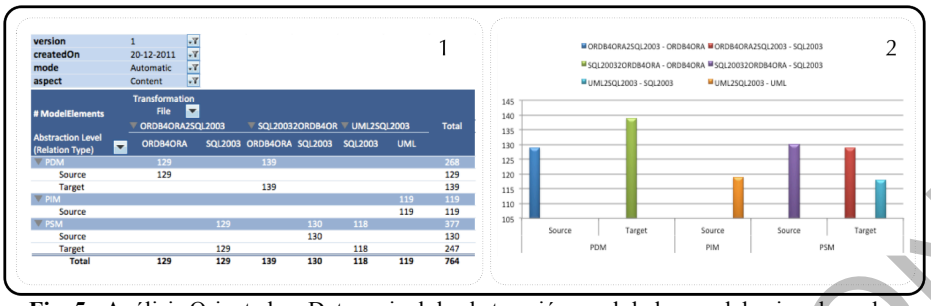

Fig. 5. Análisis Orientado a Datos: nivel de abstracción y rol de los modelos involucrados

Por ejemplo, la figura muestra que existen 129 enlaces de traza generados por la transformación ORDB4ORA2SQL2003 que apuntan a elementos del modelo ORDB4ORA con nivel de abstracción PDM y que son el origen de un enlace de traza.

Finalmente, la parte derecha de la figura (2) proporciona una representación gráfica agregada de los mismos datos. Esta representación es más conveniente para obtener una rápida visión general de los valores más destacados. Por ejemplo, observamos que el número objetos del modelo ORDB4ORA que son referenciados por trazas generadas por la transformación SQL20032ORDB4ORA destaca sobre el resto (barra vertical verde en la figura).

5 Limitaciones y trabajos futuros

En este trabajo hemos presentado un primer prototipo de un entorno para soportar el análisis de información de trazabilidad en proyectos de DSDM. Una vez comprobada la viabilidad de la propuesta, pasamos a evaluar las principales limitaciones del prototipo actual y la forma de abordar las futuras mejoras.

Aunque iTrace esta diseñado para obtener información de trazabilidad a partir de transformaciones modelo a modelo y modelo a texto, la versión actual solamente trabaja con modelos de *weaving* y transformaciones modelo a modelo. Por lo tanto, el siguiente paso es soportar la generación automática de enlaces de traza a partir de transformaciones modelo a texto. En principio, esta funcionalidad no debería entrañar una excesiva complejidad ya que existen trabajos, como de Oldevik y Neple [21] que ya la soportaban, aunque aún no hemos evaluado sus resultados.

Otro objetivo perseguido con iTrace es ofrecer soporte para consultar diferentes versiones de un mismo modelo de trazas. Disponer de diferentes versiones de un mismo modelo de traza permitiría controlar la evolución de la información de trazabilidad a lo largo del proyecto, simplificando algunas tareas como por ejemplo el análisis de huérfanos.

En cuanto al Análisis Orientado a Modelos, en la sección 3.4 se mencionaba que, de entre las diferentes formas de consultar modelos (de traza) que existían, se ha optado por utilizar EMF Query debido a que era la propuesta más madura para la consulta de modelos EMF. Sin embargo, se señalaban algunas limitaciones respecto a sus capacidades de navegación que, de alguna manera, limitan el tipo de consultas que pueden ser ejecutadas. Por lo tanto, estamos evaluando otros motores de consulta

como EMF Query2 y el uso de lenguajes imperativos para gestión de modelos como Epsilon Object Language (EOL) [37].

Respecto al Análisis Orientado a Datos, aunque actualmente utilizamos tablas dinámicas de MS Excel para soportarlo, la idea es construir el componente de Análisis Orientado a Datos sobre un almacén de datos para incrementar la potencia y alcance del análisis a la manera de los sistemas de inteligencia de negocio.

Por último, la visualización actual de los modelos iTrace (tanto el normalizado como el desnormalizado) se basa en la utilización del editor básico en forma de árbol proporcionado por EMF. Sin embargo, creemos que una representación multi-panel similar a la ofrecida por AMW [22] se ajusta mejor a la naturaleza "relacional" de los modelos de traza, cuya finalidad es representar relaciones entre los elementos de dos o más modelos. Para ello, ya hemos desarrollado un primer prototipo de un editor multi-panel que soporta la creación de enlaces de traza mediante *drag & drop* de los elementos de los modelos trazados.

6 Conclusiones

La llegada de la IDM [5] proporciona un nuevo escenario que puede ayudar a hacer realidad las ventajas que la adecuada gestión de la trazabilidad aportaría al desarrollo de software [2]. El hecho de que los modelos sean los principales artefactos manipulados a lo largo del proceso de desarrollo y de que estén conectados mediante transformaciones de modelos permite derivar semiautomáticamente las trazas entre los diferentes artefactos [5].

Sin embargo, el estudio de la literatura existente pone de manifiesto la falta de propuestas centradas en un uso adecuado de la información de trazabilidad que puede ser recogida de cualquier proyecto de DSDM. Además, las pocas propuestas que abordan el tema no consideran los diferentes tipos de actores involucrados en el proceso de desarrollo. Por ejemplo, las necesidades de un desarrollador de transformaciones de modelos pueden ser diferentes a las necesidades de un jefe de proyecto.

Para abordar estos problemas, este trabajo ha presentado iTrace, un framework para soportar el análisis de información de trazabilidad automáticamente generada a partir de un proyecto de DSDM. Más concretamente, iTrace soporta dos tipos de análisis, llamados Análisis Orientado a Modelos y Análisis Orientado a Datos.

El primero está orientado a aquellos actores que poseen un perfil más operacional y su principal característica es que los resultados se proporcionan en forma de modelos que, a su vez, pueden ser procesados utilizando técnicas de IDM, tales como las transformaciones de modelos [9] para filtrar la información que contienen.

Por otra parte, el objetivo del Análisis Orientado a Datos es agregar información de bajo nivel para proporcionar datos que pueden ser usados para dar soporte al proceso de toma de decisiones. Por ejemplo, un jefe de proyecto puede usar el análisis mostrado en el caso de estudio para identificar si el número trazas que apunta a los objetos de un determinado modelo es excepcionalmente elevado. Si así fuera, podemos inferir que el impacto de cualquier modificación realizada sobre dicho modelo afectará a muchos otros artefactos del proyecto. En tal caso, pudiera convenir descomponer

dicho modelo en modelos más pequeños que redujesen el impacto de cualquier modificación.

Finalmente, nos gustaría hacer hincapié en el hecho de que iTrace es solamente un prototipo preliminar. A pesar de ser completamente funcional, todavía deja mucho margen de mejora. En este sentido, este trabajo puede ser visto como un *position paper* que ha servido para comprobar la viabilidad de la propuesta y para identificar las principales áreas de trabajo futuro. Consideramos que los resultados son prometedores y ya estamos trabajando en el desarrollo de las diferentes mejoras.

Agradecimientos

Este trabajo se ha llevado a cabo en el marco del proyecto MASAI (Ref. TIN-2011-22617) financiado por el Ministerio de Ciencia y Tecnología de España.

Referencias

1. IEEE Standard Glossary of Software Engineering Terminology. IEEE Std 610.12-1990. 1 (1990).
2. Ramesh, B., Stubbs, C., Powers, T., Edwards, M.: Requirements traceability: Theory and practice. Annals of Software Engineering. 3, 397-415 (1997).
3. Asuncion, H., Asuncion, A., Taylor, R.: Software traceability with topic modeling, (2010).
4. Hayes, J.H., Dekhtyar, A., Sundaram, S.K.: Advancing candidate link generation for requirements tracing: the study of methods. Software Engineering, IEEE Transactions on. 32, 4-19 (2006).
5. Bezivin, J.: In search of a basic principle for model driven engineering. Novatica Journal, Special Issue. (2004).
6. Stahl, T., Voelter, M., Czarnecki, K.: Model-Driven Software Development: Technology, Engineering, Management. John Wiley & Sons (2006).
7. Santiago, I., Jiménez, Á., Vara, J.M., Castro, V.D., Bollati, V.A., Marcos, E.: Model-Driven Engineering As a New Landscape For Traceability Management: A Systematic Review. Information and Software Technology (Submitted). (2012).
8. Aizenbud-Reshef, N., Nolan, B.T., Rubin, J., Shaham-Gafni, Y.: Model traceability. IBM Systems Journal. 45, 515-526 (2006).
9. Bernstein, P.: Applying model management to classical meta data problems. First Biennial Conference on Innovative Data Systems Research. , Asilomar, CA, USA (2003).
10. Anquetil, N., Kulesza, U., Mitschke, R., Moreira, A., Royer, J.C., Rummler, A., Sousa, A.: A model-driven traceability framework for software product lines. Software and Systems Modeling. 9, 427-451 (2010).
11. Olsen, G.K., Oldevik, J.: Scenarios of traceability in model to text transformations, (2007).
12. Walderhaug, S., Johansen, U., Stav, E., Aagedal, J.: Towards a generic solution for traceability in MDD. 2th European Conference on Model Driven Architecture - Traceability Workshop (ECMDA-TW'06). (2006).
13. Sánchez, P., Alonso, D., Rosique, F., Álvarez, B., Pastor, A.: Introducing Safety Requirements Traceability Support in Model-Driven Development of Robotic Applications. IEEE Transactions on Computers. 60, 1059-1071 (2011).
14. Valderas, P., Pelechano, V.: Introducing requirements traceability support in model-driven development of web applications. Information and Software Technology. 51, 749-768 (2009).
15. Paige, R., Olsen, G.K., Kolovos, D., Zschaler, S., Power, C.: Building model-driven engineering traceability classifications. Proceedings of the 4th European Conference on Model Driven Architecture - Traceability Workshop (ECMDA-TW'08). pp. 49-58. , Berlin, Germany (2008).

16. Tisi, M., Cabot, J., Jouault, F.: Improving higher-order transformations support in ATL. International Conference on Model Transformation (ICMT) (2010).
17. Jouault, F.: Loosely coupled traceability for ATL, (2005).
18. Sanders, G., Shin, S.: Denormalization Effects on Performance of RDBMS. Proceedings of the 34th Annual Hawaii International Conference on System Sciences (HICSS-34)-Volume 3 - Volume 3. p. 3013--. IEEE Computer Society, Washington, DC, USA (2001).
19. Steinberg, D., Budinsky, F., Paternostro, M., Merks, E.: EMF: Eclipse Modeling Framework. Addison-Wesley Professional (2008).
20. Jouault, F., Allilaire, F., Bézivin, J., Kurtev, I.: ATL: A model transformation tool. Science of Computer Programming. 72, 31-39 (2008).
21. Oldevik, J., Neple, T., Grønmo, R., Aagedal, J., Berre, A.-J.: Toward Standardised Model to Text Transformations. In: Hartman, A. and Kreische, D. (eds.) Model Driven Architecture – Foundations and Applications. pp. 239-253. Springer Berlin / Heidelberg (2005).
22. Didonet del Fabro, M., Bézivin, J., Valduriez, P.: Weaving Models with the Eclipse AMW plugin. Eclipse Modeling Symposium. (2006).
23. Jouault, F., Bézivin, J., Kurtev, I.: TCS: a DSL for the specification of textual concrete syntaxes in model engineering. Proceedings of the 5th international conference on Generative programming and component engineering. pp. 249-254. ACM, New York, NY, USA (2006).
24. The AMMA Platform, http://www.sciences.univ-nantes.fr/lina/atl/AMMAROOT/.
25. Bézivin, J., Jouault, F., Rosenthal, P., Valduriez, P.: Modeling in the Large and Modeling in the Small. In: Aßmann, U., Aksit, M., and Rensink, A. (eds.) Model Driven Architecture. p. 901. Springer Berlin / Heidelberg (2005).
26. The Eclipse Project: MDT OCL, http://www.eclipse.org/modeling/mdt/?project=.
27. The Eclipse Project: EMF Model Query, http://www.eclipse.org/modeling/emf/?project=query.
28. The Eclipse Project: EMF Model Query 2, http://www.eclipse.org/modeling/emf/?project=query2.
29. Clasen, C., Jouault, F., Cabot, J.: VirtualEMF: A Model Virtualization Tool. In: De Troyer, O., Bauzer Medeiros, C., Billen, R., Hallot, P., Simitsis, A., and Van Mingroot, H. (eds.) Advances in Conceptual Modeling. Recent Developments and New Directions. pp. 332-335. Springer Berlin / Heidelberg (2011).
30. Mernik, M., Heering, J., Sloane, A.M.: When and how to develop domain-specific languages. ACM Computing Surveys (CSUR). 37, 316-344 (2005).
31. Watson, H.J., Goodhue, D.L., Wixom, B.H.: The benefits of data warehousing: why some organizations realize exceptional payoffs. Information & Management. 39, 491-502 (2002).
32. Kimball, R.: The Data Warehouse Lifecycle Toolkit. Wiley (1998).
33. Inmon, W.H.: Building the Data Warehouse. Wiley (1996).
34. Vara, J., Vela, B., Bollati, V., Marcos, E.: Supporting Model–Driven Development of Object–Relational Database Schemas: A Case Study. In: Paige, R. (ed.) Theory and Practice of Model Transformations. pp. 181-196. Springer Berlin / Heidelberg (2009).
35. Vara, J.M., Castro, M.V.D., Fabro, M.D.D., Marcos, E.: Using Weaving Models to automate Model-Driven Web Engineering proposals. International Journal of Computer Applications in Technology. 39, 245-252 (2010).
36. Feuerlicht, G., Pokorný, J., Richta, K.: Object-Relational Database Design: Can Your Application Benefit from SQL:2003? In: Barry, C., Lang, M., Wojtkowski, W., Conboy, K., and Wojtkowski, G. (eds.) Information Systems Development. pp. 975-987. Springer (2009).
37. Kolovos, D., Paige, R., Polack, F.: The Epsilon Object Language (EOL). In: Rensink, A. and Warmer, J. (eds.) Model Driven Architecture – Foundations and Applications. pp. 128-142. Springer Berlin / Heidelberg (2006).