Gestión de la trazabilidad en el desarrollo dirigido por modelos de Transformaciones de Modelos: una revisión de la literatura

Álvaro Jiménez, Juan M. Vara, Verónica A. Bollati, Esperanza Marcos

Grupo de Investigación Kybele,
Departamento de Lenguajes y Sistemas Informáticos II,
Universidad Rey Juan Carlos, Madrid (España).
{alvaro.jimenez,juanmanuel.vara,
veronica.bollati,esperanza.marcos}@urjc.es

Resumen La Ingeniería Dirigida por Modelos ofrece un escenario ideal para potenciar el papel de la trazabilidad en el desarrollo de software. El hecho de que los modelos y las transformaciones pasen a ocupar un papel principal permite contemplar las trazas como enlaces entre los elementos de dichos modelos que podrían a su vez representarse en otros modelos y por tanto transformarse, generarse, etc. Por otro lado, como artefactos software que son, resultaría deseable aplicar los principios de la Ingeniería Dirigida por Modelos a su desarrollo. Combinando estas dos ideas, este trabajo realiza una revisión de la literatura para estudiar cómo se lleva a cabo la gestión de la trazabilidad en el desarrollo dirigido por modelos de transformaciones de modelos. El objetivo pasa por identificar si existen propuestas que sean capaces de incluir mecanismos para la gestión de la trazabilidad en la transformación desarrollada.

Palabras Clave: Desarrollo Dirigido por Modelos, Trazabilidad, Transformaciones de Modelos, Estado del Arte.

1. Introducción

La IEEE [17] define la trazabilidad como:

> «*El grado de relación que puede establecerse entre dos o más productos de un proceso de desarrollo, especialmente productos que tienen relaciones predecesor-sucesor o maestro-subordinado con otro producto.*»

Gestionar la trazabilidad permite conocer cómo evolucionan los elementos del sistema a lo largo del proceso de desarrollo y cómo se relacionan entre sí [2]. Esta información puede utilizarse en diferentes actividades como el análisis de impacto del cambio, la toma de decisiones de diseño o el mantenimiento [24]. Así, dada la importancia de la trazabilidad en el desarrollo de los sistemas, sería deseable que las metodologías de desarrollo dieran soporte para la gestión de la trazabilidad [32].

Por otro lado, la Ingeniería Dirigida por Modelos (IDM) es el último paso en la tendencia a elevar el nivel de abstracción al que se diseña y construye software. Los principios fundamentales de la IDM son: potenciar el rol de los modelos a lo largo de todo el ciclo de vida y potenciar el nivel de automatización en el proceso de desarrollo [31]. La aplicación de los principios de la IDM al desarrollo de software da lugar al Desarrollo de Software Dirigido por Modelos (DSDM) [37]. En un proceso de DSDM se parte de modelos con un alto nivel de abstracción que servirán para especificar el sistema, obviando detalles tecnológicos. Dichos modelos son refinados hasta que su nivel de abstracción permita utilizarlos para generar el código final. El nexo que une cada nuevo paso son las transformaciones modelo-a-modelo (M2M), por tanto su rol en el proceso es tan importante como el de los propios modelos. Dado que estamos en el contexto de la IDM y que las transformaciones también son artefactos software, resultaría recomendable aplicar los principios de la IDM a su desarrollo. Si se definen las transformaciones como un modelo más, podremos generarlas, transformarlas o validarlas como cualquier otro modelo del sistema [5][6]. En este sentido, existen algunas propuestas como [4], [6] o [35] para el desarrollo dirigido por modelos (DDM) de transformaciones de modelos.

Dada la importancia de la gestión de la trazabilidad en el desarrollo software y las ventajas que ofrece en este sentido la IDM y en concreto, las transformaciones de modelos resulta interesante disponer de un entorno de DDM de transformaciones de modelos que soporte la generación y gestión de las trazas existentes entre los elementos implicados en la transformación. Con el objetivo de identificar si existen propuestas que pongan en práctica esta idea y analizar la forma en que lo hacen, hemos realizado una revisión sistemática cuyo proceso se basa en el propuesto por Pino *et al.* en [28]. En este trabajo presentamos los principales resultados obtenidos en esta revisión de la literatura así como las líneas de trabajo futuras, surgidas como resultado de dicho estudio.

El resto del documento se encuentra estructurado de la siguiente forma: en la Sección 2 se presentan los diferentes criterios de clasificación y evaluación definidos para llevar a cabo el análisis de cada una de las propuestas. La Sección 3 analiza las propuestas tratadas en el estado del arte, de acuerdo a los criterios de evaluación y clasificadas según las categorías definidas en la sección anterior. La Sección 4 presenta una discusión basada en la evaluación de los diferentes trabajos analizados. Por último, en la Sección 5 se obtienen las conclusiones derivadas a partir del estado del arte y se identifican líneas de trabajo futuras.

2. Marco de Evaluación

El objetivo de esta revisión de la literatura es identificar la existencia de propuestas para el DDM de transformaciones de modelos que ofrezcan soporte para la gestión de la trazabilidad. Para obtener una visión global de la literatura, la revisión se ha realizado identificando dos categorías: por un lado, las propuestas orientadas al DDM de transformaciones de modelos y por otro, las propuestas para la gestión de la trazabilidad en el desarrollo de transformaciones. Las

primeras se caracterizan por adoptar una aproximación generativa para el desarrollo de transformaciones, proponiendo la especificación a distintos niveles y la (semi)automatización del paso de un nivel a otro. Las segundas tratan de integrar soporte para gestionar la trazabilidad en la propia transformación, que se desarrolla siguiendo la aproximación tradicional, es decir, manualmente y desde cero. Dado que los objetivos son diferentes, no es posible emplear los mismos criterios para la evaluación de las propuestas incluidas en cada grupo. Por este motivo, definimos un conjunto de criterios o características de evaluación para cada grupo o categoría. Estos críterios han sido establecidos durante el proceso de revisión sistemática mencionado, a partir de un análisis inicial de las propuestas. Los criterios definidos para cada una de las categorías son los siguientes:

Propuestas para el desarrollo dirigido por modelos de transformaciones de modelos. Incluye las propuestas existentes para el desarrollo de transformaciones de modelos siguiendo una aproximación dirigida por modelos. Los criterios o características a evaluar para estas propuestas serán:

1. **Nivel de abstracción en la especificación de la transformación**. Indica los niveles de abstracción a los que se permite especificar la transformación. Nos centraremos en evaluar si soportan especificaciones independientes del lenguaje de transformación y especificaciones dependientes del lenguaje de transformación.
2. **Soporte para la trazabilidad**. Evalúa si la propuesta ofrece mecanismos de soporte para la gestión de la trazabilidad.
3. **Soporte tecnológico**. Analiza si la propuesta ha sido implementada o si por el contrario, se limita a ofrecer una solución teórica.

Propuestas para la gestión de la trazabilidad en el desarrollo de transformaciones de modelos. Incluye las propuestas que presentan métodos o actividades para dar soporte a la gestión de la trazabilidad en el contexto del desarrollo de transformaciones. Para estas propuestas los criterios o características a evaluar serán:

1. **Nivel de abstracción de la transformación que soporta trazabilidad**. Determina el nivel de abstracción de la transformación a partir del cual se ofrece soporte para la gestión de la trazabilidad, por ejemplo a nivel de metatransformación o a nivel de transformación para un lenguaje específico.
2. **Generación de las relaciones de trazabilidad**. Permite conocer si las relaciones de trazabilidad entre los elementos son definidas automáticamente a partir de la transformación o si es necesario que sean definidas por el usuario.
3. **Metamodelo de trazabilidad**. Evalúa si la propuesta define un metamodelo genérico de trazabilidad o si emplea metamodelos específicos para cada escenario.
4. **Gestión de las trazas**. Determina la forma de almacenar las trazas obtenidas: en un modelo de trazas, embebidas en los propios modelos del sistema, en un repositorio de trazas, etc.

5. **Soporte tecnológico**. Al igual que en la categoría anterior, evalúa si la propuesta ha sido implementada.

3. Revisión de la literatura

En esta sección se presenta la revisión de la literatura, de acuerdo a las categorías de clasificación y criterios de evaluación indicados en la sección anterior.

3.1. Propuestas para el desarrollo dirigido por modelos de transformaciones de modelos

En esta sección se analizan las características identificadas para las propuestas orientadas al desarrollo dirigido por modelos de transformaciones de modelos.

Nivel de abstracción en la especificación de la transformación. La mayoría de las propuestas analizadas de acuerdo a este criterio proponen el desarrollo de transformaciones de modelos a distintos niveles de abstracción. Entre estas propuestas, podemos destacar los trabajos de Bézivin *et al.* [4], de Küster *et al.*[22] y de Vignaga [36].

Bézivin *et al.* proponen en [4] realizar el modelado de las transformaciones mediante especificaciones independientes de plataforma (*PIT, Platform Independent Transformation*) y específicas de plataforma (*PST, Platform Specific Transformation*). Siguiendo con esta línea, en [5] se propone definir modelos de transformación conformes a metamodelos PIT y a partir de ellos, obtener transformaciones para distintas plataformas de implementación.

En [35], Vignaga propone una metodología para aplicar técnicas de IDM al desarrollo y evolución de las transformaciones de modelos y en [36], propone realizar la especificación de las transformaciones siguiendo una estructura de cuatro niveles de abstracción. Esta estructura parte del nivel más bajo, que corresponde al código que implementa la transformación en una plataforma concreta. El segundo nivel suprime los detalles específicos de la plataforma. En el tercer nivel se especifican las relaciones entre los elementos y en el cuarto nivel se incluyen mecanismos de modularización.

En [22], Küster *et al.* presentan un método sistemático para la construcción de transformaciones de modelos, centrado en el diseño de las mismas. Proponen realizar la distinción entre el diseño de las transformaciones de alto nivel, de bajo nivel y, por último, la validación del diseño de las transformaciones.

Sin embargo, otros autores como Didonet del Fabro, en su tesis doctoral [9], desarrollan propuestas para el modelado de transformaciones en un lenguaje específico. En este caso, una de las principales contribuciones de su propuesta consiste en la definición de modelos de *Weaving* para realizar transformaciones de modelos y posteriormente generar la transformación en lenguaje ATL (*ATLAS Transformation Language*, [19]).

Soporte para la trazabilidad. Entre las propuestas analizadas, ninguna de ellas ofrece soporte para la gestión de la trazabilidad. Tan sólo, Didonet del Fabro [9] indica que es posible emplear modelos de *Weaving* para la representación de las relaciones de trazabilidad entre los elementos de diferentes modelos.

Soporte tecnológico. La mayoría de los trabajos no ofrecen una implementación de su propuesta teórica. Sólo Didonet del Fabro [9] presenta una herramienta para validarla. Concretamente *ATLAS Model Weaver* (AMW) sirve para establecer y representar relaciones entre elementos de distintos modelos. AMW ha sido implementado como extensión del entorno de desarrollo Eclipse. Por otro lado, Bézivin *et al.* [4] indican que están trabajando en la implementación de un conjunto de herramientas que den soporte a las ideas presentadas en el marco del programa de investigación CARROL. Sin embargo, actualmente este proyecto parece abandonado ya que la web del proyecto no se encuentra disponible.

3.2. Propuestas para la gestión de la trazabilidad en el desarrollo de transformaciones de modelos

Esta segunda categoría engloba propuestas para el soporte de la gestión de la trazabilidad en el desarrollo de transformaciones de modelos. A diferencia de las propuestas anteriores, estas no se centran en aplicar los principios de la IDM al desarrollo de las transformaciones sino que tienen por objetivo dar soporte para la generación de trazas a partir de la ejecución de transformaciones. En esta sección analizamos las características identificadas para estas propuestas:

Nivel de abstracción de la transformación que soporta trazabilidad. La mayor parte de los trabajos analizados proponen gestionar la trazabilidad a nivel de transformación. Esto es, proponen un método para la generación de trazas a partir de una transformación de modelos, definida para un lenguaje de transformación concreto. Por ejemplo, Bondé *et al.* [7] proponen un método basado en ModTransf [12], un motor de transformaciones basado en reglas XML; Jouault [18] gestiona la trazabilidad a partir de transformaciones ATL; Levendovszky *et al.* [23] emplean GReAT [3], una aproximación basada en grafos, para las transformaciones de modelos; los autores Olsen y Oldevik [25] presentan un método basado en MOFScript, un lenguaje de transformaciones modelo a texto (M2T); Sánchez *et al.* [30] proponen soporte de trazabilidad a partir de transformaciones ATL (M2M) y JET (M2T); finalmente Valderas y Pelechano [33] proponen un método basado en transformaciones de grafos implementadas en AGG (*Attributed Graph Grammars*).

Dos casos particulares son los trabajos de Boronat *et al.* [8] y Guerra *et al.* [16]. Los primeros proponen la definición de operadores algebraicos sobre los modelos para llevar a cabo diferentes tareas propias de la IDM, entre las que se encuentra la gestión de la trazabilidad. Por su parte, Guerra *et al.* proponen una aproximación similar, pero basada en patrones declarativos para definir operaciones entre los modelos. Estas propuestas no están ligadas a un lenguaje

de transformación concreto, por tanto, podemos decir que presentan un mayor nivel de abstracción en cuanto a la gestión de la trazabilidad.

Generación de las relaciones de trazabilidad. Para llevar a cabo la generación de las trazas a partir de una transformación de modelos, es necesario que dicha transformación conozca las relaciones de trazabilidad existentes entre los elementos de dichos modelos. Para definir estas relaciones existen dos aproximaciones: generación automática y definición manual.

La generación automática también es conocida como generación implícita porque las trazas son generadas a partir de las relaciones entre elementos que recogen las reglas de la transformación, sin necesidad de especificar explícitamente que existe una relación de trazabilidad. En el caso de seguir una aproximación manual, el usuario deberá incluir de forma explícita las relaciones de trazabilidad en la especificación de la transformación. Es decir, en este caso, además de especificar que un elemento a se transforma en un elemento b, es necesario indicar que existe una relación de trazabilidad entre los elementos a y b. En la literatura es posible encontrar ejemplos de ambas aproximaciones e incluso algunas propuestas que plantean una solución que combina ambas.

Entre los trabajos que siguen la aproximación automática se encuentran [7], [8], [16] y [25]. En estos trabajos podemos analizar tres formas diferentes de aplicar esta aproximación. En [7], Bondé *et al.* proponen modificar el funcionamiento de ModTransf [12] para obtener automáticamente las relaciones de trazabilidad a partir de las reglas de la transformación. Boronat *et al.* [8] definen operadores algebraicos sobre los modelos para llevar a cabo diferentes tareas, así cuando se ejecuta una transformación, las relaciones de trazabilidad se obtienen automáticamente a partir de dichos operadores. Guerra *et al.* [16] plantean el uso de patrones declarativos para llevar a cabo diferentes operaciones entre los modelos y las relaciones de trazabilidad son obtenidas a partir de estos patrones. Los autores Olsen y Oldevik [25], presentan una propuesta basada en MOFScript. Ya que MOFScript gestiona la trazabilidad de forma interna a partir de la transformación, los autores aprovechan esta funcionalidad para generar un modelo de trazas.

Entre las propuestas que plantean una definición manual de las relaciones de trazabilidad se encuentran Levendovszky *et al.* [23], Sánchez *et al.* [30] y Valderas y Pelechano [33]. De estas propuestas, [23] y [33] coinciden en el método para definir manualmente las relaciones de trazabilidad: proponen incluir las relaciones de trazabilidad directamente en las reglas de la transformación. Sánchez *et al.* presentan un método donde el usuario crea el primer modelo de trazas, un modelo de los requisitos de seguridad y una arquitectura acorde a ellos. A partir de esta información y mediante sucesivas transformaciones ATL (M2M) y JET (M2T), se generan de forma automática los siguientes modelos de trazas. Esta aproximación puede considerarse (semi-)automática, sin embargo ya que parte de un modelo de trazas definido manualmente, consideramos que sigue la aproximación manual.

Además de las aproximaciones anteriores, debemos destacar el trabajo presentado por Jouault en [18]. Este trabajo propone incluir las relaciones de trazabilidad en las reglas de la transformación, al igual que los trabajos [23] y [33], ya mencionados. Sin embargo, su diferencia estriba en que además, Jouault presenta una herramienta basada en una transformación ATL que permite incluir relaciones de trazabilidad de forma automática en las transformaciones.

Otra aproximación interesante es la que plantean Grammel y Kastenholz en [15]. Presentan una interfaz genérica llamada GTI (*Generic Traceability Interface*) que sirve como conexión con diferentes mecanismos de transformaciones de modelos. GTI permite obtener de forma automática las relaciones de trazabilidad que son generadas en la transformación, ya sea de forma implícita o explícita. Por tanto, depende directamente de cómo realice la gestión de la trazabilidad el motor de transformación empleado.

Metamodelo de trazabilidad. Para llevar a cabo el modelado de las trazas es necesario disponer de un metamodelo de trazabilidad [1]. En este contexto existen dos aproximaciones: emplear un metamodelo de trazabilidad de propósito general o emplear un metamodelo de trazabilidad específico para cada escenario.

Drivalos *et al.* presentan en [10] un estudio donde evalúan las dos posibilidades. Según estos autores, la primera aproximación facilita la interoperabilidad entre los modelos de trazas pero en algunos casos no permite ajustarse a la realidad del escenario, ya que los modelos de trazas resultan demasiado genéricos. Por el contrario, los metamodelos específicos permiten generar modelos que se ajustan mejor a escenarios concretos, sin embargo implican un mayor esfuerzo, ya que es necesario definir un metamodelo para cada escenario. Además, como consecuencia de emplear distintos metamodelos, genera problemas de interoperabilidad entre modelos de distintos escenarios. Finalmente, Drivalos *et al.* defienden el empleo de metamodelos de trazabilidad específicos por ser más rigurosos a la hora de definir las trazas.

Sin embargo, la mayoría de los trabajos analizados proponen el empleo de un metamodelo de trazabilidad de propósito general. Dado que para contemplar cualquier escenario de trazabilidad deben ser lo suficientemente genéricos, la mayor parte de los autores proponen metamodelos que permiten modelar los elementos presentes en una transformación de modelos: elementos del modelo y relaciones entre los elementos. De estos metamodelos, podemos destacar el propuesto por Bondé *et al.* en [7], ya que se trata de un metamodelo más detallado que permite además, registrar las operaciones (creación, enlace, copia, conversión y transformación) que dan lugar a la generación de la traza.

Gestión de las trazas. Una vez que las trazas han sido generadas, es necesario almacenarlas para su posterior tratamiento o análisis. Al igual que en las características anteriores, podemos distinguir diversas aproximaciones. Las más comunes son: almacenar las trazas embebidas en los propios modelos del sistema o crear nuevos modelos que contengan las trazas, es decir, modelos de trazas.

En [20], los autores realizan un estudio de las ventajas e inconvenientes de ambas aproximaciones. Incluir las trazas en los modelos del sistema *contamina* los modelos, sin embargo, facilita la visualización y el análisis por parte de observadores humanos ya que toda la información se encuentra localizada en un mismo contenedor. La segunda aproximación evita los problemas de contaminación, pero dificulta la visualización y el análisis. Dado que ambas aproximaciones tienen ventajas e inconvenientes, los autores plantean una solución intermedia: generar modelos externos que contengan las trazas y posteriormente, fusionarlos (*merging*) con los modelos del sistema, obteniendo modelos «a la carta». A pesar de proponer esta solución intermedia, en trabajos posteriores como [10], defienden el empleo de modelos de trazas que eviten la contaminación de los modelos.

Así mismo, la mayoría de las propuestas analizadas para la generación de trazas a partir de transformaciones de modelos emplean la aproximación basada en crear modelos de trazas. La idea más común en la literatura es generar como resultado de la transformación el/los modelo/s de salida y un modelo que contenga las trazas. Además, este modelo de trazas es conforme al metamodelo de trazabilidad definido. Este tipo de aproximación podemos encontrarla en [7], [8], [18] o [25].

Un caso diferente a los anteriores lo podemos encontrar en Levendovszky *et al.* [23]. En este caso, las trazas son almacenadas en dos repositorios: uno para las trazas generadas a partir de transformaciones M2M y otro para las generadas mediante transformaciones M2T. Además de almacenarse por separado, esta información también es representada de formas distintas. En el primer caso son representadas mediante grafos y en el segundo de forma textual.

Soporte tecnológico. Al contrario que las propuestas analizadas en la Sección 3.1, la mayor parte de los trabajos analizados en esta categoría sí disponen de una implementación de su propuesta teórica. De hecho, las únicas propuestas que no presentan herramienta de soporte son las presentadas por Bondé *et al.* [7] y por Drivalos *et al.* [10][20], aunque esta segunda propuesta introduce TML (*Traceability Metamodelling Language*) [11] un lenguaje para la construcción de metamodelos de trazabilidad.

El resto de trabajos han desarrollado una herramienta de soporte o han sido implementadas sobre herramientas ya existentes. La mayoría de ellas, especialmente las primeras, han sido desarrolladas sobre Eclipse, probablemente el marco de trabajo más empleado en el desarrollo de herramientas dirigidas por modelos [34]. Así, encontramos herramientas como MOMENT (*MOdel manageMENT*)[8], un *framework* basado en el uso de operadores algebraicos para gestionar modelos; PAMOMO [16], una herramienta para la definición y ejecución de patrones declarativos sobre los modelos; o *TraceAdder* [18], que utiliza una transformación ATL para añadir información en modelos de transformación ATL para la generación de modelos de trazas.

Los trabajos que ofrecen una implementación sobre herramientas ya existentes son aquellos como [15] que presenta GTI, una interfaz de conexión con

los motores de transformación y, por tanto, depende directamente de la implementación del motor concreto. Otros trabajos que se incluyen en este grupo son [25], [30] y [33], ya que para llevar a cabo las transformaciones emplean marcos de trabajo ya existentes como el *framework* para transformaciones de modelos proporcionado por ATL y su aportación consiste proporcionar herramientas para analizar y visualizar las trazas generadas. Por ejemplo, [30] presenta *Safety-Requirements-Trace-Tool (SRTT)*, una herramienta para la generación de informes de trazabilidad en formato HTML.

4. Discusión

Una vez presentada la revisión de la literatura, esta sección presenta las principales conclusiones extraídas de la evaluación de los trabajos analizados, de acuerdo a los criterios definidos en la Sección 2 para cada categoría de clasificación. Estas conclusiones se resumen en las Tablas 1 y 2, respectivamente.

Autores	Nivel de Especificación de la Transformación	Soporte para la Trazabilidad	Soporte Tecnológico (*)
Bézivin *et al.*	Indep. / Dep.	No	–
Didonet del Fabro	Dependiente	No	**
Küster *et al.*	Indep. / Dep.	No	–
Vignaga	Indep. / Dep.	No	–

(*) Leyenda	
Símbolo	Valor
–	No presenta
*	Emplea otras Herramientas
**	Herramienta propia

Tabla 1. Clasificación de propuestas para el desarrollo dirigido por modelos de transformaciones de modelos

La Tabla 1 muestra la evaluación de las *propuestas para el desarrollo dirigido por modelos de transformaciones de modelos*. Como se puede observar, la mayor parte de estas propuestas, a excepción de Didonet del Fabro [9], permiten la definición de transformaciones a distintos niveles de abstracción. Sin embargo, ninguna soporta la gestión de la trazabilidad a partir de las transformaciones definidas, por lo que no cumplen el objetivo buscado en esta revisión. También es necesario destacar que el único trabajo que ofrece una herramienta que implementa la propuesta es precisamente [9]. Por tanto, parece que no es una tarea trivial ofrecer soporte tecnológico para el desarrollo de transformaciones a diferentes niveles de abstracción.

La Tabla 2 resume las *propuestas para la gestión de la trazabilidad en el desarrollo de transformaciones de modelos*. En este caso podemos observar que la mayoría coinciden en la forma de gestionar las trazas generadas (emplear modelos de trazas) y en el tipo de metamodelo de trazabilidad empleado (metamodelo de propósito general). Estas aproximaciones evitan la *contaminación* de los modelos del sistema y dado que emplean un mismo metamodelo para todos los escenarios, facilitan la interoperabilidad entre los modelos de trazas.

[1] Los valores empleados en la columna «Soporte Tecnológico» de la Tabla 2 se corresponden con los empleados en el mismo criterio de la Tabla 1

Autores	Soporte de Trazabilidad a nivel de...	Generación de las Relaciones	Metamodelo	Gestión de las trazas	Soporte Tecnológico[1]
Bondé *et al.*	Transformación	Automática	General	Modelo de Trazas	–
Boronat *et al.*	Op. Algebraicos	Automática	Gen./Espec.	Modelo de Trazas	**
Grammel y Kastenholz	Transformación	Auto./Manual	General	Modelo de Trazas	*
Guerra *et al.*	Patrones Declarativos	Automática	General	Modelo de Trazas	**
Jouault	Transformación	Auto./Manual	General	Modelo de Trazas	**
Kolovos/Drivalos *et al.*	Modelos (No transf.)	Manual	Específicos	Modelo de Trazas	–
Levendovszky *et al.*	Transformación	Manual	General	Grafo/Texto	**
Olsen y Oldevik	Transformación	Automática	General	Modelo de Trazas	*
Sánchez *et al.*	Transformación	Manual	General	Modelo de Trazas	*
Valderas y Pelechano	Transformación	Manual	General	Modelo de Navegación	*

Tabla 2. Clasificación de propuestas para la gestión de la trazabilidad en el desarrollo de transformaciones de modelos

Para la validación de las propuestas presentadas, a diferencia de la categoría anterior, la mayor parte de los autores han desarrollado su propia herramienta o se basan en el uso de otras herramientas ya existentes. Tan sólo Bondé *et al.* y Kolovos/Drivalos *et al.* no presentan implementación.

En cuanto a la generación de las relaciones de trazabilidad siguen diversas aproximaciones: generación automática, definición manual o una combinación de ambas. En el contexto del DSDM, es deseable soportar el mayor nivel de automatización posible. Sin embargo, las tareas automatizadas no siempre proporcionan los resultados esperados por el desarrollador. Por ejemplo, puede que sólo se requiera la generación de un conjunto limitado de trazas y no de todas ellas para un objetivo concreto. Por tanto, la aproximación ideal sería aquella que genere las trazas de forma automática, pero permita que los usuarios refinen el resultado del proceso de generación.

Por último evaluamos el nivel de soporte que se ofrece para la gestión de la trazabilidad. La mayoría de las propuestas analizadas permiten la generación de información de trazabilidad a partir de la definición de transformaciones dependientes de un lenguaje de transformación concreto, pero tan sólo en los trabajos [8] y [15] se presentan soluciones que dan soporte para la gestión de la trazabilidad a niveles independientes del lenguaje que implementa la transformación.

5. Conclusiones y Líneas Futuras

La gestión de la trazabilidad en el desarrollo de software ha sido un problema recurrente en la Ingeniería del Software en las últimas décadas [2]. Con la llegada de la IDM y el DSDM, los modelos y las transformaciones que los conectan se convierten en los principales artefactos en cualquier actividad relacionada con el desarrollo de software. Este escenario proporciona un marco ideal para abordar mejoras y nuevas propuestas para la gestión de la trazabilidad. Además, en este contexto, resulta recomendable aplicar los principios de la IDM también al desarrollo de las transformaciones [5]. Por todo ello, resultaría interesante la posibilidad de incorporar mecanismos para la gestión (semi-)automática de la trazabilidad en el DDM de transformaciones de modelos. Así, en este trabajo

hemos presentado una revisión de la literatura centrada en identificar y analizar cómo se lleva a cabo la gestión de la trazabilidad en el DDM de transformaciones de modelos.

Como se puede desprende de las conclusiones presentadas en la Sección 4, las propuestas para el DDM de transformaciones de modelos no ofrecen soporte para la gestión de la trazabilidad y las propuestas para la gestión de la trazabilidad en el desarrollo de transformaciones de modelos no contemplan el modelado de estas transformaciones y por extensión, no aplican los principios de la IDM a su desarrollo. Por tanto, podemos concluir que no se ha encontrado ninguna propuesta que satisfaga plenamente la generación de trazas en el desarrollo dirigido por modelos de transformaciones de modelos.

Así, dado que en anteriores trabajos hemos abordado el desarrollo semi-automático de metatransformaciones de modelos [6], una contribución respecto al estado del arte sería mejorar la propuesta para el desarrollo dirigido por modelos, de forma que las transformaciones generadas incorporen, además, los mecanismos necesarios para generar las trazas entre los elementos implicados en la transformación. Es decir, los objetivos finales serán: 1) Soportar el modelado de la transformación a alto nivel. 2) Generar modelos de transformación de más bajo nivel que incorporen los elementos necesarios para soportar la trazabilidad. 3) Generar el código que implementa la transformación a partir de dichos modelos. De esta forma, además del modelo destino, la transformación generará las trazas entre los elementos del modelo origen y el modelo destino de la transformación. 4) Definir un metamodelo de trazabilidad para permitir el modelado de las trazas.

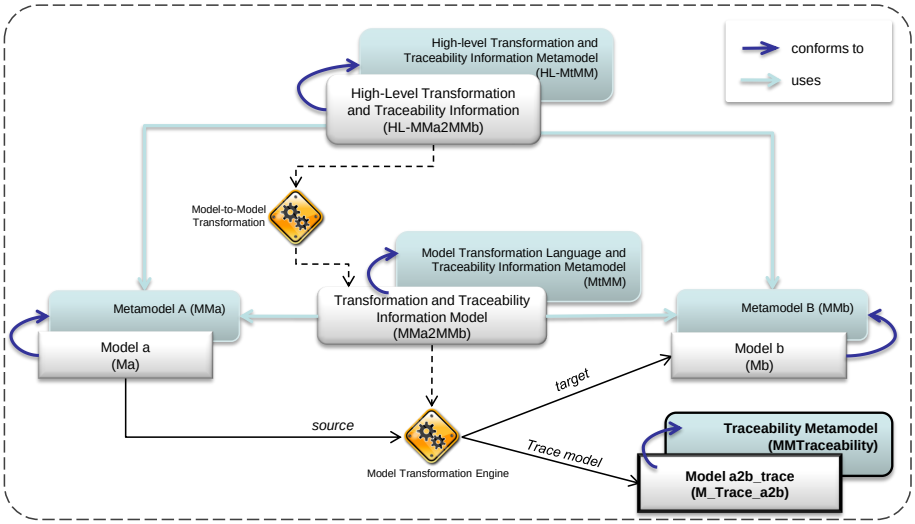

Figura 1. Proceso de Transformación de Modelos y Generación de Modelo de Trazas

Teniendo en cuenta estos objetivos, la Figura 1 muestra una adaptación del proceso de transformación de modelos presentado en [6]. Este proceso soportará el DDM de transformaciones de modelos incorporando soporte para la gestión de la trazabilidad entre los elementos de los modelos implicados en la transformación.

Para realizar la transformación entre dos modelos (`Ma y Mb`), conformes a los metamodelos origen (`MMa`) y destino (`MMb`), se define un conjunto de relaciones entre sus elementos. Estas relaciones son recogidas, sin tener en cuenta el lenguaje que implementará la transformación, en un modelo de transformación de alto nivel de abstracción (`HL-MMa2MMb`). A continuación debe ser transformado en un modelo de bajo nivel (`MMa2MMb`), conforme al metamodelo de un lenguaje de transformación concreto (`MtMM`). Este modelo refina las relaciones entre los elementos de los metamodelos (`MMa y MMb`), de acuerdo a las particularidades del motor de transformación del lenguaje escogido y por tanto, este modelo puede ser directamente traducido al código que implementa la transformación en dicho lenguaje.

Se propone además, incorporar información de trazabilidad en los modelos intermedios con el objetivo de generar las trazas entre los elementos de los modelos origen y destino (`Ma y Mb`). De esta forma, la transformación además de generar el modelo de salida (`Mb`), generará un modelo de trazas (`M_Trace_a2b`), conforme a un metamodelo genérico de trazabilidad (`MMTraceability`).

Agradecimientos. Este trabajo se ha realizado en el marco de los proyectos: Model-Caos (TIN2008-03582), Agreement-Technologies (CSD2007-0022) financiados por el Ministerio Español de Educación y Ciencia; y, el subprograma de Personal Técnico de Apoyo (MICINN-PTA - 2009), perteneciente al Programa Nacional De Contratación e Incorporación de RRHH cofinanciado por el Ministerio Español de Ciencia e Innovación.

Referencias

1. Aizenbud-Reshef, N., Nolan, B. T., Rubin, J., Shaham-Gafni, Y. (2006); Model traceability. IBM System Journal, vol. 45(3), pp. 515-526.
2. Asuncion, H. (2008); Towards practical software traceability. 30th international conference on Software engineering, ICSE 2008.
3. Balasubramanian, D., Narayanan, A., van Buskirk, C. P., Karsai, G. (2006); The Graph Rewriting and Transformation Language: GreAT. ECEASST.
4. Bézivin, J., Farcet, N., Jezequel, J.M., Langlois, B., Pollet, D. (2003); Reflective Model Driven Engineering. UML, vol. 2863 of LNCS, pp. 175-189.
5. Bézivin, J., Büttner, F., Gogolla, M., Jouault, F., Kurtev, I., Lindow, A. (2006); Model Transformations? Transformation Models!. 9th International Conference on Model Driven Engineering Languages and Systems, MoDELS 2006, Genova, Italia.
6. Bollati, V. A. (2011); MeTAGeM: un Entorno de Desarrollo de Transformaciones de Modelos Dirigido por Modelos. Tesis Doctoral, Universidad Rey Juan Carlos, Madrid.

7. Bondé L., Boulet, P., Dekeyser, J. L. (2005); Traceability and interoperability at different levels of abstraction in model transformations. Forum on Specification and Design Languages (FDL'05). Laussane, Suiza.
8. Boronat, A., Carsí, J. A., Ramos, I. (2005); Automatic support for traceability in a generic model management framework. 1st European Conference on Model-Driven Architecture Foundations and Applications (ECMDA). Nuremberg, Alemania.
9. Didonet Del Fabro, M. (2007); Metadata management using model weaving and model transformation. Tesis Doctoral, Universidad de Nantes, Francia.
10. Drivalos, N., Paige, R. F., Fernandes, K. J., Kolovos, D. S. (2008); Towards rigorously defined model-to-model traceability. 4th ECMDA Traceability Workshop. Berlin, Alemania.
11. Drivalos, N., Kolovos, D. S., Paige, R. F., Fernandes, K. J. (2009); Engineering a DSL for software traceability. Software Language Engineering, LNCS vol. 5452/2009, pp. 151-167. Springer-Link.
12. Dumoulin, C. (2004); ModTransf: A model to model transformation engine.
13. Falleri, J., Huchard, M., Nebut, C. (2006); Towards a traceability framework for model transformations in kermeta. 2nd ECMDA Traceability Workshop. Bilbao.
14. Galvao, I., Goknil, A. (2007); Survey of traceability approaches in model-driven engineering. 11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007). Annapolis (Maryland), Estados Unidos.
15. Grammel, B., Kastenholz, S. (2010); A generic traceability framework for facet-based traceability data extraction in model-driven software development. Proceedings of the 6th ECMFA Traceability Workshop. París, Francia.
16. Guerra, E., de Lara, J., Kolovos, D. S., Paige, R. F. (2010); Inter-modelling: From theory to practice. Model Driven Engineering Languages and Systems, LNCS vol. 6394/2010, pp. 376-391. Ed. Springer-Link.
17. IEEE (1990); IEEE Standard Glossary of Software Engineering Terminology. IEEE Press, Piscataway.
18. Jouault, F. (2005); Loosely coupled traceability for ATL. 1st ECMDA Workshop on Traceability. Nuremberg, Alemania.
19. Jouault, F., Kurtev, I. (2006); Transforming Models with ATL. Satellite Events at the MoDELS 2005 Conference, pp. 128-138.
20. Kolovos, D. S., Paige, R. F., Polack, F. A. C. (2006); On-demand merging of traceability links with models; 2th ECMDA Traceability Workshop. Bilbao, España.
21. Kolovos, D., Paige, R., Polack, F. (2008); The Epsilon Transformation Language. 1st International Conference on Model Transformation, Zurich.
22. Küster, J. M., Ryndina, K., R. Hauser (2005); A Systematic Approach to Designing Model Transformations. Report RZ 3621, IBM, Zurich, July 2005.
23. Levendovszky, T., Balasubramanian, D., Smyth, K., Shi, F., Karsai, G. (2010); A transformation instance-based approach to traceability. 6th ECMFA Traceability Workshop. París, Francia.
24. Naslavsky, L., Alspaugh, T. A., Richardson, D. J., Ziv, H. (2005); Using Scenarios to Support Traceability. 3rd International Workshop on Traceability in Emerging Forms of Software Engineering (TEFSE'05), pp. 25-30. ACM Press.
25. Olsen, G., Oldevik, J. (2007); Scenarios of traceability in model to text transformations. 3rd European Conference on Model-Driven Architecture Foundations and Applications (ECMDA). Haifa, Israel.
26. OMG (2003); MDA Guide Version 1.0.1. Documento: omg/2003-06-01.
27. Paige, R. F., Drivalos, N., Kolovos, D. S., et al. (2010); Rigorous identification and encoding of trace-links in model-driven engineering. Software and Systems Modeling, SpringerLink.

28. Pino, F., Gracía, F., Piattini, M. (2008); Software process improvement in small and medium software enterprises: a systematic review. Software Quality Journal, vol. 16(2), pp. 237-261.
29. Ramesh, B., Jarke, M. (2001); Towards reference models for requirements traceability. IEEE Transactions on Software Engineering, vol. 27, Issue 1.
30. Sánchez, P., Alonso, D., Rosique, F., Alvarez, B.,Pastor, J. (2010); Introducing safety requirements traceability support in model-driven development of robotic applications. IEEE Transactions on Computers. IEEE Computer Society.
31. Schmidt, D. C. (2006); Model Driven Engineering. IEEE Computer, Vol. 39(2), pp. 41 - 47.
32. Spanoudakis, G., Zisman, A. (2005); Software Traceability. Handbook of Software Engineering and Knowledge Engineering, vol. III: Recent Advancements.
33. Valderas, P., Pelechano, V. (2009); Introducing requirements traceability support in model-driven development of web applications. Information and Software Technology, vol. 51(4), pp. 749-768. Elsevier.
34. Vara, J. M. (2009); M2DAT: A Technical Solution for Model-Driven Development of Web Information Systems. Tesis Doctoral, Universidad Rey Juan Carlos, 2009.
35. Vignaga, A. (2007); A methodological approach to developing model transformations. Model-Driven Engineering Languages and Systems (MoDELS 2007). Nashville (TN), Estados Unidos.
36. Vignaga, A. Perovich, D., Bastarrica, M.C. (2007); Towards Layered Specifications of Model Transformation. Reporte Técnico.
37. Voelter, M. (2009); Best Practices for DSLs and Model-Driven Development. Journal of Object Technology, vol. 8, no. 6, September-October 2009, pp. 79 - 102.