risti
Revista Ibérica de Sistemas e Tecnologias de Informação
Revista Ibérica de Sistemas y Tecnologías de Información

Recebido / Recibido: 24/10/2010
Aceitação / Aceptación: 03/12/2010

M2DAT-HYMO: una herramienta basada en MDA para la generación automática de aplicaciones Web a partir del modelo del hipertexto

Feliu Trias Nicolau, Iván Santiago Viñambres, Juan Manuel Vara, Valeria de Castro

feliu.trias@urjc.es, ivan.santiago@urjc.es, juanmanuel.vara@urjc.es, valeria.decastro@urjc.es

Grupo de Investigación Kybele, Departamento de Lenguajes y Sistemas Informáticos II, Universidad Rey Juan Carlos, Avda. Tulipán s/n, 28933, Móstoles (Madrid)

Resumen: Una premisa básica exigible a cualquier propuesta metodológica para el desarrollo de software es la de ofrecer algún tipo de herramienta de apoyo. En el caso de propuestas basadas en los principios de la Ingeniería Dirigida por Modelos (MDE) esta premisa se convierte en un requisito imprescindible. Este trabajo presenta la herramienta que soporta HM3, el método para el modelado del hipertexto del marco metodológico MIDAS. Así, el artículo presenta la definición y codificación de las transformaciones automáticas entre los modelos que componen HM3, y las transformaciones modelo a texto que soportan la generación automática de código. En este sentido, uno de los aspectos más innovadores de este trabajo es la utilización de modelos weaving para personalizar algunas de las transformaciones entre modelos que propone HM3.

Palabras clave: MDA, MDE, Transformaciones de Modelos, Modelos Weaving, Generación de Código

Abstract: Nowadays, any software development methodological proposal must be accompanied by the corresponding technological support. Such premise becomes even more relevant when the proposal is based on Model Driven Engineering principles. Therefore, this proposal presents the tool support for HM3, the Hypertext Modelling Method of MIDAS. In particular, it presents the specification and coding of the model-to-model transformations that bridge the different models that compose HM3, as well as the model-to-text transformations that support code generation. One of the most outstanding aspects of this proposal is the use of weaving models to personalize the model-to-model transformations proposed by HM3.

Keywords: MDA, MDE, Model Transformations, Weaving Models, Code Generation

1. Introducción

Los Sistemas de Información Web (SIW) se han convertido en los últimos años en una herramienta clave, no sólo para el funcionamiento diario, sino también para la expansión y crecimiento de cualquier organización: la mayor parte de su actividad, comercial y de gestión se realiza a través de Internet y mediante estos sistemas (Lowe, 2003). Por este motivo, en la actualidad adquieren especial relevancia las técnicas y metodologías que facilitan la construcción de este tipo de sistemas.

Por otra parte, uno de los paradigmas de desarrollo software más importante en la actualidad es la Ingeniería Dirigida por Modelos (MDE, *Model Driven Engineering*) (Selic, 2001; Schmidt, 2006) cuyo principio fundamental es trasladar el foco de atención de las actividades de codificación a las actividades de modelado. De este modo, los modelos pasan a ser los artefactos principales en el proceso de desarrollo.

Dentro de este nuevo paradigma surge MIDAS (Cáceres et al., 2006; Marcos, et al., 2006; De Castro, 2007), una metodología dirigida por modelos para el desarrollo de aplicaciones orientadas a servicios (De Castro et al., 2009) que considera los tres aspectos fundamentales asociados con la clásica arquitectura de tres capas: contenido, hipertexto y comportamiento. Como parte de la propuesta de MIDAS se ha definido un método específico para el desarrollo de cada capa. En particular, para la capa del hipertexto MIDAS propone un Método para el Modelado del Hipertexto (HM^3, *Hypertext Modelling Method of MIDAS*). Este método propone un conjunto de modelos a distintos niveles de abstracción y las reglas de transformación entre ellos. En particular se adopta la separación en niveles de abstracción propuesta por la Arquitectura Dirigida por Modelos (MDA, *Model Driven Architecture*) (Selic, 2001; Miller & Mukerji, 2003) que distingue entre Modelos Independientes de Computación (CIMs, *Computer Independent Models*), Modelos Independientes de Plataforma (PIMs*, Platform Independent Models*) y Modelos Específicos de Plataforma (PSMs, *Platform Specific Models*).

El soporte tecnológico para la propuesta de MIDAS es M2DAT (MIDAS MDA Tool) un entorno extensible y modular, de forma que las herramientas de soporte para cada propuesta metodológica comprendida en MIDAS se desarrollan como un nuevo módulo que se integra con el resto de la herramienta (Vara, 2009). Este artículo se centra en M2DAT-HYMO (M2DAT *for HYpertext MOdelling*), el módulo de M2DAT que da soporte a HM^3. En particular se presentan los editores gráficos para los diferentes modelos propuestos en HM^3; las transformaciones modelo-a-modelo (M2M) para conectar dichos modelos y las transformaciones modelo-a-texto (M2T) para generar el código que implementa el sistema a partir de dichos modelos.

En este punto conviene mencionar que un problema de las herramientas que dan soporte al desarrollo dirigido por modelos de SIWs, como ArgoUWE (Knapp et al., 2003), OOWS Suite (Valverde et al., 2007) o HyperDE (Hypermedia Developing Environment) (Nunes & Schwabe, 2006) gira en torno el objetivo de soportar una automatización completa del proceso de desarrollo propuesto. No siempre los modelos son capaces de recoger toda la información necesaria para generar el código que implementa el sistema. En otras palabras, no siempre se pueden recoger todas las decisiones de diseño en las transformaciones que conectan los diferentes modelos. Por ello, frecuentemente es necesario llevar refinar los modelos generados por una

transformación concreta. Para resolver este problema, en este trabajo se propone la utilización de modelos de weaving para soportar la personalización de la forma en que una transformación genérica se ejecuta.

Este artículo está organizado de la siguiente manera: la sección 2 presenta el Método de Modelado del Hipertexto de MIDAS describiendo los modelos que lo componen; la sección 3 explica el proceso de desarrollo de la herramienta M2DAT-HYMO; finalmente, la sección 4 presenta los trabajos relacionados y la sección 5 las principales conclusiones y las líneas para el trabajo futuro.

2. El Método del Modelado del Hipertexto de MIDAS: HM3

HM3, el método para el modelado del hipertexto de MIDAS, propone la utilización de 5 modelos diferentes: el modelo de casos de uso, el modelo de casos de uso extendido, el modelo de fragmentos extendido, el modelo de navegación extendido y el modelo conceptual de datos (Cáceres et al., 2006).

Como se puede observar en la Figura 1, el marco metodológico de MIDAS contempla 3 aspectos en el desarrollo del SIW: contenido, hipertexto y comportamiento. Aunque HM3 está enfocado en el aspecto del hipertexto utiliza modelos pertenecientes a los aspectos de contenido –modelo conceptual de datos– y comportamiento –modelo de casos de uso y modelo de casos de uso extendido–, todos ellos definidos a nivel PIM.

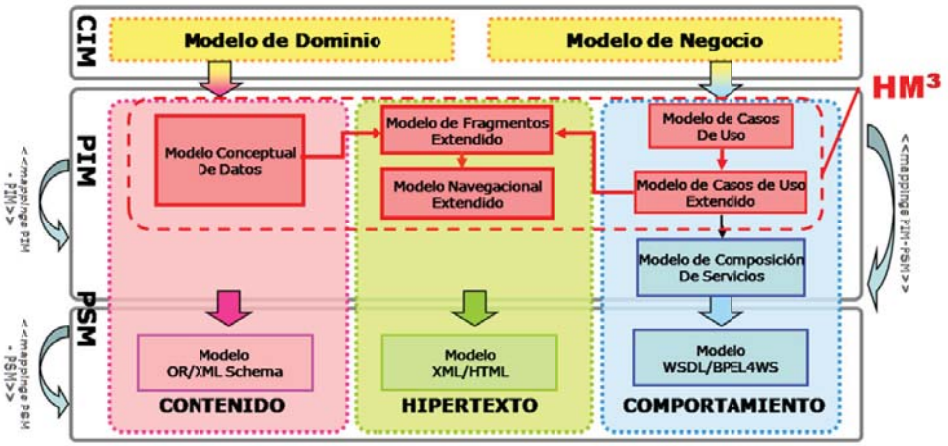

Figura 1– Detalle del proceso de HM3 en el marco de MIDAS

A partir del modelo de casos de uso, que recoge la funcionalidad del sistema, se obtienes sucesivamente y de forma semi-automática el modelo de casos de uso extendido, el modelo de fragmentos extendido y finalmente el modelo de navegación extendido. El contenido de los nodos que definen esta estructura navegacional se obtiene a partir del modelo conceptual de datos. Por último, a partir del modelo de navegación extendido se genera el código que implementa parte del SIW.

El objetivo de este último modelo es representar la estructura navegacional del SIW. De esta forma, podemos identificar las funcionalidades que el sistema soporta y los pasos que debe seguir el usuario para utilizar cada una de ellas. El conjunto de estos pasos se

denomina ruta principal (*main route)* y es una de las principales aportaciones de HM3. En particular, mejora la navegabilidad del sistema porque permite guiar al usuario en la ejecución de cualquier servicio ofrecido por el SIW. En el contexto de la Ingeniería Web, la usabilidad se asocia tradicionalmente con la navegación (Palmer, 2002). Por lo tanto, un sistema de navegación que permita a los usuarios encontrar la información buscada de forma rápida y eficiente contribuye a mejorar la usabilidad del SIW, aumentando sus probabilidades de éxito (Nielsen & Loranger, 2006).

3. Soporte Tecnológico para HM3: M2DAT-HYMO

En las siguientes secciones se presentan las distintas actividades relacionadas con la construcción de M2DAT-HYMO utilizando un caso de estudio: un SIW para una comunidad científica compuesta de varios grupos de investigación que permite consultar la información relativa a los proyectos de investigación que gestiona.

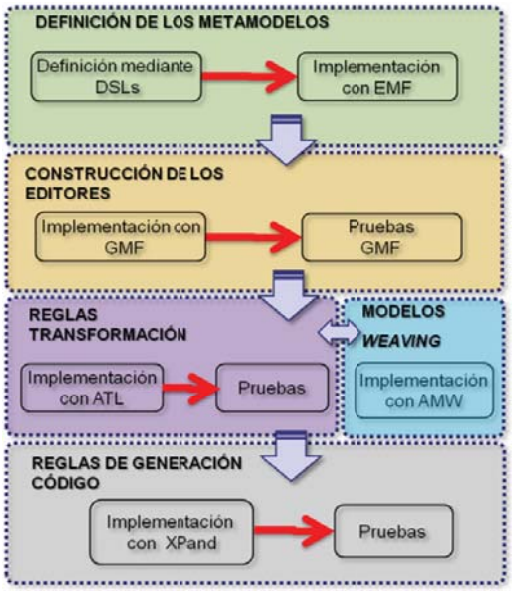

Figura 2– Proceso de desarrollo de M2DAT-HYMO

La Figura 2 representa un esquema de las tareas realizadas y el orden en el que se han llevado a cabo. Desde la implementación y definición de los metamodelos –sección 3.1–, pasando por el desarrollo de editores gráficos –sección 3.2–, la implementación de las transformaciones de modelos –sección 3.3– y los modelos de weaving –sección 3.4– hasta llegar a la generación de código –sección 3.5–.

3.1 Definición e implementación de los metamodelos

El soporte tecnológico para cualquier propuesta metodológica en el marco de la MDE comprende las herramientas necesarias para trabajar con un conjunto de lenguajes específicos de dominio (DSLs, Domain Specific Languages) (Mernik et al., 2005). El primer paso hacia la construcción de cada uno de estos DSLs pasa por definir e implementar el metamodelo subyacente.

Para la definición e implementación de los metamodelos en M2DAT-HYMO se ha utilizado EMF (*Eclipse Modelling Framework*) (Budinsky et al., 2008), un framework de Eclipse que permite definir, editar y manejar metamodelos. Posibilita la implementación en Ecore –el lenguaje de metamodelado de EMF– de los metamodelos de los DSLs de HM3. A partir del metamodelo se pueden generar herramientas para la edición y gestión de modelos conformes a dicho metamodelo.

En los últimos años EMF se ha convertido en el marco de metamodelado más utilizado en el contexto de la MDE (Vara, 2009). Por lo tanto, utilizar este framework para implementar los DSLs de M2DAT-HYMO aumenta el nivel de interoperabilidad de los modelos de HM3 ya que podrán ser importados y/o exportados desde/hacia otras herramientas MDE basadas en EMF (Vara, 2009). Otra ventaja es que EMF es un framework abierto que incorpora periódicamente nuevas tecnologías que surgen en el contexto de la MDE, lo que aumenta la extensibilidad de cualquier herramienta construida sobre EMF.

A partir de la implementación de los metamodelos en EMF se generó de forma semi-automática un editor gráfico para cada modelo de HM3. El principal inconveniente de estos editores fue la compleja estructura jerárquica con la que representaban los modelos. Por este motivo se optó por la construcción de editores gráficos de tipo nodos y arcos (o diagramadores) que simplificasen la representación gráfica de los modelos. El desarrollo de éstos se explica en la sección 3.2.

3.2 Desarrollo de editores gráficos

Tras la implementación de los metamodelos en EMF de los distintos DSLs se pasó a construir los diagramadores para éstos. Para esta tarea se contemplaron dos alternativas: utilizar un GPL (*Graphical Programming Language*) o utilizar una herramienta generativa, que proporcionase parte del código que implementaba dichos editores, como GMF (*Graphical Modelling Framework*) el framework de modelado gráfico de Eclipse (Eclipse Foundation, 2008).

El uso de un GPL permite construir editores gráficos más optimizados y potentes desde un punto de vista gráfico, pero supone una mayor inversión de recursos porque obliga a codificar prácticamente desde cero cada funcionalidad de la que se desea dotar al editor. Por otra parte, GMF permite generar de forma semi-automática el código que implementa los editores gráficos para cada uno de los DSLs, a partir de los metamodelos implementados en EMF. Además, este framework permite especificar el aspecto (forma, tamaño, color, etc.) de cada uno de los elementos del modelo que van a ser representados con el editor y definir cómo será la paleta de herramientas del editor gráfico.

Por todo esto se eligió GMF como herramienta para la construcción de los editores gráficos. La Figura 3 muestra, a modo de ejemplo, el editor gráfico que permite representar el modelo de casos de uso extendido. Por motivos de espacio, en este trabajo no presentamos el resto de editores, pero el proceso de desarrollo para ellos ha sido el mismo.

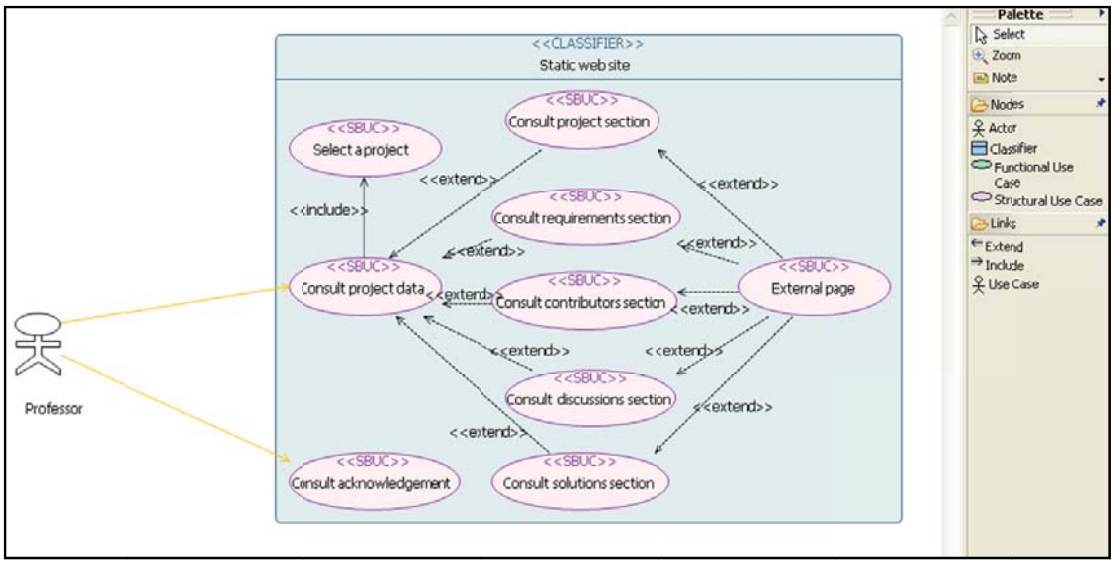

Figura 3– Editor gráfico implementado en GMF para el modelo de casos de uso extendido

3.3 Implementación de las transformaciones de modelos

Uno de los principios básicos de la MDE es la automatización del proceso de desarrollo. Por lo tanto, el siguiente paso en el desarrollo de M2DAT-HYMO fue implementar las reglas de transformación que conectan los diferentes modelos que componen en proceso de desarrollo propuesto.

Las alternativas evaluadas para el desarrollo de estas transformaciones fueron las siguientes: propuestas basadas en el uso de gramáticas de grafos (como MOLA o VIATRA); lenguajes de transformación que implementan el estándar Query/View/Transformation (QVT), tanto declarativos (MediniQVT o ModelMof) como imperativos (SmartQvt o Borland Qvto); el lenguaje de transformación ATL (Jouault & Piers, 2009).

Se optó por utilizar ATL por ser un lenguaje estable y maduro, aunque en constante fase de mejora y del cual se puede encontrar documentación variada –manuales, escenarios– y herramientas de comunicación, como listas de noticas o foros muy activos que ayudan a resolver cualquier duda no cubierta por la documentación (Vara, 2009). Otro motivo importante a la hora de escoger ATL fue su buen acoplamiento con la herramienta AMW (*ATLAS Model Weawer*) (Didonet Del Fabro et al., 2006), lo que permite utilizar modelos de weaving para establecer relaciones entre los diferentes modelos utilizados a lo largo del proceso, así como para implementar modelos de anotación. Estos modelos permiten representar la información adicional que permite llevar introducir decisiones de diseño en las transformaciones sin perjudicar el nivel de automatización de la propuesta.

En este trabajo se han definido e implementado dos transformaciones. La que permite pasar del modelo de casos de uso extendido al modelo de fragmentos extendido (EUCM2ESM) y la que, a partir de este último, genera el modelo de navegación extendido (ESM2ENM). Por motivos de espacio, nos centraremos en la transformación EUCM2ESM. Esta transformación EUCM2ESM toma como entrada el modelo de casos

de uso extendido, el modelo de weaving que implementa el modelo de anotación y el diagrama de clases UML que representa el modelo conceptual de datos.

Las reglas de transformación se definen siempre a nivel de metamodelo. De esta forma, cada regla especifica un patrón origen y un patrón destino en base a los metamodelo origen y destino. Cada vez que se encuentre una coincidencia entre el patrón origen y el modelo origen, la regla producirá en el modelo destino el patrón destino. A modo de ejemplo, la Figura 4 muestra la implementación en ATL de una de las reglas de transformación definidas en el modelo EUCM2ESM. Esta regla convierte un elemento *Functional Use Case* del modelo de casos de uso extendido en un elemento *Functional Slice* del modelo de fragmentos extendido.

```
rule FunctionalUseCase2FunctionalSlice
{
        from
                u : EUCM!FunctionalBasicUseCase
        to
                fs : ESM!FunctionalSlice (model <-thisModule.model,name <-u.name,
                sliceProperties <-u.getSliceProperties ()
                ->collect(p|thisModule .UmlProperty2SliceProperty (p)).debug ('SliceProperties'))
}
```

Figura 4 – Regla de transformación en ATL

3.4 Uso de modelos de weaving

Como se mencionaba en la introducción, uno de los aspectos tecnológicos de esta propuesta es la utilización de modelos de weaving para poder dirigir la ejecución de las diferentes transformaciones de modelos.

La Figura 5 muestra de forma esquemática el uso habitual de un modelo de weaving: establecer o identificar relaciones entre los elementos de dos o más modelos. Por ejemplo, el objeto R_1 del modelo de weaving relaciona el elemento A del modelo M_A con los elementos Y y Z del modelo M_B.

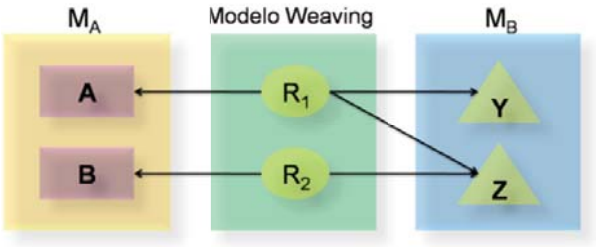

Figura 5 – Utilización habitual de un modelo de weaving

Los modelos de weaving también se definen conforme a un metamodelo implementado en Ecore. Esta es otra de las ventajas de haber usado EMF como marco de metamodelado: tanto los metamodelos de los DSLs de M2DAT-HYMO, como los metamodelos para los modelos de weaving e incluso el metamodelo de ATL o el de UML están definidos usando un mismo lenguaje de metamodelado, y esto aumenta la interoperabilidad entre los modelos. La Figura 6 describe esta idea.

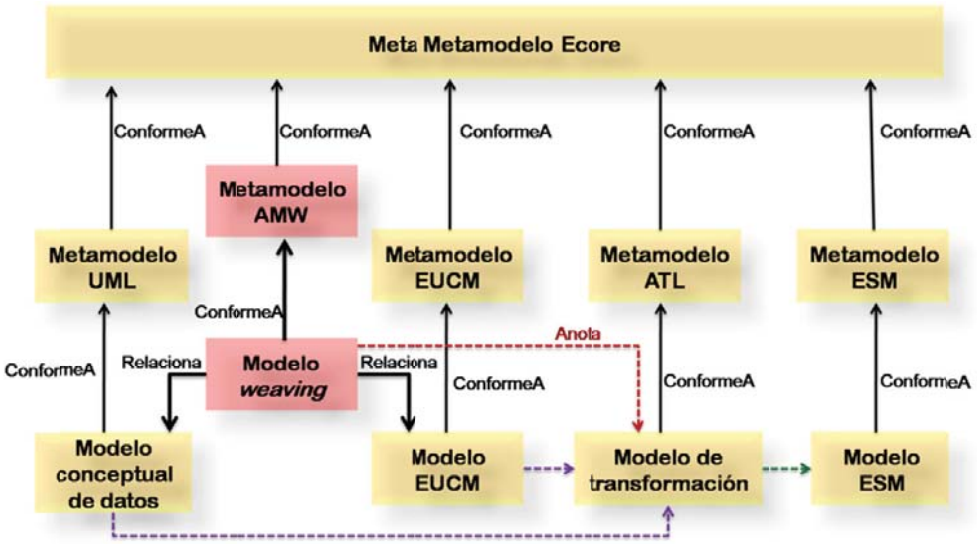

Figura 6– Relación entre modelos y metamodelos en M2DAT-HYMO

El modelo de weaving relaciona los elementos del modelo de casos de uso extendido (EUCM, *Extended Use Case Model*) con los elementos del modelo conceptual de datos. También se puede observar que el modelo de casos de uso extendido, el modelo conceptual de datos y el modelo de weaving son las entradas de la transformación. Los dos últimos proporcionan información adicional para guiar la transformación del modelo de casos de uso extendido en el modelo de fragmentos extendido (ESM, *Extended Slice Model*).

Para adaptar los modelos de weaving a las necesidades de cada transformación utilizamos el mecanismo de extensión soportado por AMW. Así, definimos un nuevo metamodelo, de forma que en los modelos definidos conformes a dichos metamodelos se puedan incluir los objetos que representan la información adicional que dirige la ejecución de la transformación. La Figura 7 recoge dicho metamodelo, donde se resaltan los elementos añadidos. Éstos son: los elementos *Containment* y *Association* que derivan del elemento *WLink*, y los elementos *Container*, *Property*, *Source* y *Target* que heredan del elemento *Associated Model Element,* que a su vez proviene del elemento *WLinkEnd*.

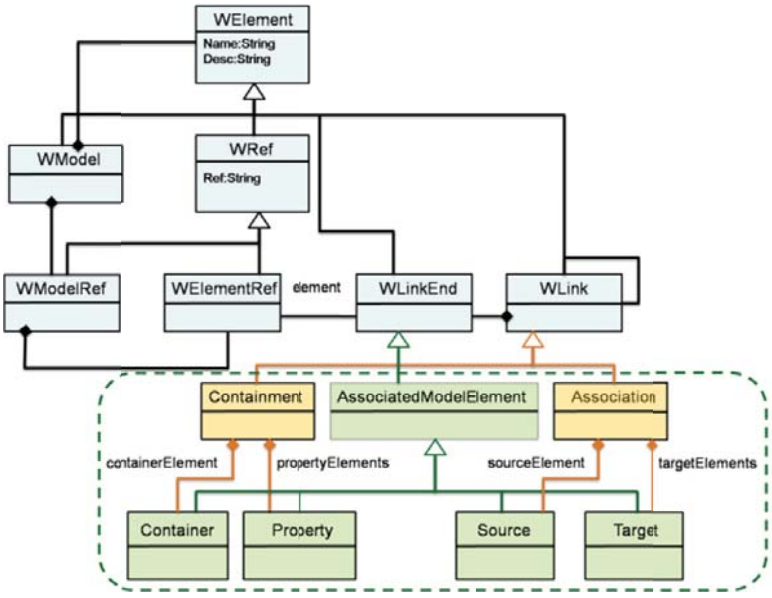

Figura 7– Extensión del metamodelo del modelo weaving

En particular, para el caso de estudio que se incluye en este artículos se han implementado dos modelos de weaving. El primero, llamado *AnnotatingCM,* relaciona los elementos *property* del modelo conceptual de datos con los elementos *basic use case* del modelo de casos de uso extendido. De esta forma, el modelo *AnnotatingCM* proporciona información para la ejecución de la transformación EUCM2ESM. Permite asociar un conjunto de propiedades a cada fragmento o *slice* del modelo de fragmentos extendido que genera. La Figura 8 muestra un sencillo ejemplo de cómo se utiliza este primer modelo de weaving.

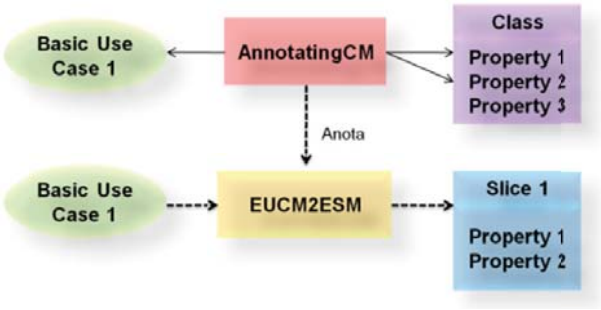

Figura 8– Utilización del modelo de weaving *AnnotatingCM*

El segundo modelo, *AnnotatingUCM*, asocia los elementos *business service* del modelo de casos de uso con los elementos *basic use case* del modelo de casos de uso extendido. A partir de los elementos *business service* se obtienen los elementos *main route* del modelo de fragmentos extendido. Una *main route* tiene asociada un conjunto de elementos *slice* involucrados en la ejecución de un servicio ofrecido por el SIW. De esta forma, este modelo de weaving aporta información para la ejecución de la transformación EUCM2ESM. Permite agrupar los elementos *slice* en las diferentes *main routes* existentes. De nuevo, la Figura 9 muestra un pequeño ejemplo de cómo se usa este modelo.

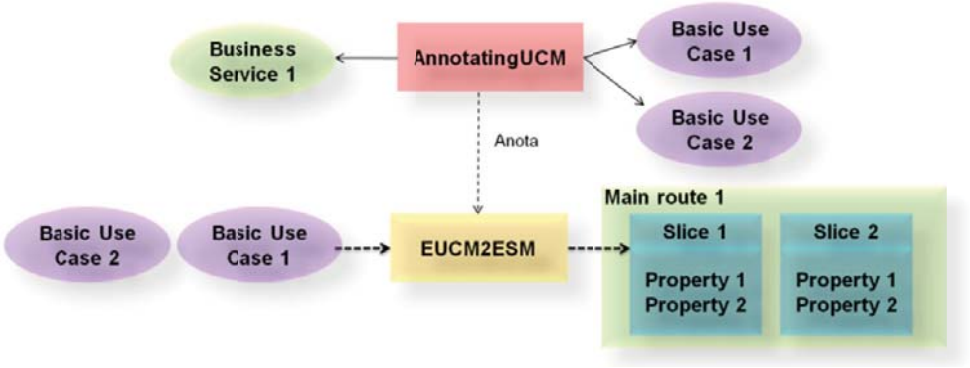

Figura 9– Utilización del modelo de anotación *AnnotatingUCM*

3.5 Generación de código

La última tarea en el desarrollo de M2DAT-HYMO fue la implementación de las reglas para la generación automática de código. Existen varias herramientas en el marco de Eclipse dirigidas a dar soporte a esta actividad, como las Java Emitter Templates (JET) (Powell, 2004) o lenguajes orientados directamente a utilizar modelos como entradas para la generación de código, como MofScript (Oldvik et al., 2005) o Xpand de openArchitectureWare (Klatt, 2007).

En particular, Xpand es un DSL basado en plantillas para la generación de código. Posee una sintaxis básica que, en combinación con Xtend (Efftinge, 2007), le permite incrementar su alcance para transformaciones modelo a modelo.

Un proyecto de generación de código basado en Xpand necesita incluir el metamodelo concreto como parte del mismo. Además, contiene el fichero de *workflow,* el fichero de propiedades del *workflow* y el fichero de configuración. El paquete *templates* contiene todos los ficheros de plantillas para la generación de código. El fichero de *workflow* representa el núcleo central de un proyecto de generación de código y permite indicar el modo deseado de ejecución de la generación de código. Tiene una estructura de componentes en el que cada uno será el encargado de una labor diferente. Se pueden distinguir los siguientes componentes: (1) el componente *xmiParser* que contiene la estructura del modelo y es donde se identifica el paquete principal del metamodelo. (2) El componente *dirCleaner* que elimina el directorio donde se deposita el código generado. (3) El componente *generator* indica el paquete del metamodelo, el nodo raíz a partir del cual vamos a empezar a generar el código y el directorio de salida para la generación de código.

Para que M2DAT-HYMO pueda generar código a partir del modelo de navegación extendido es necesario definir reglas que permitan transformar los elementos del modelo a código y así obtener fragmentos HTML, hipervínculos entre ellos y las rutas de navegación que proporcionará el sistema. Dado que HM3 no incorpora un modelo de presentación, es necesario que nuestras reglas de transformación generen otros dos tipos de contenidos. Por una parte, una hoja de estilos que permita dar formato a la salida del código HTML generado; y por otra parte un fichero de funciones Javascript que proporcione la funcionalidad necesaria para manipular la información de las rutas principales. Con ello dotamos al SIW de una interfaz gráfica aceptable y cierto comportamiento.

A modo de ejemplo, la Figura 10(a) muestra el mapa de navegación generado por M2DAT-HYMO para el caso de estudio, referido a la ruta principal *consult a project*. La Figura 10(b) muestra cada uno de los ficheros generados.

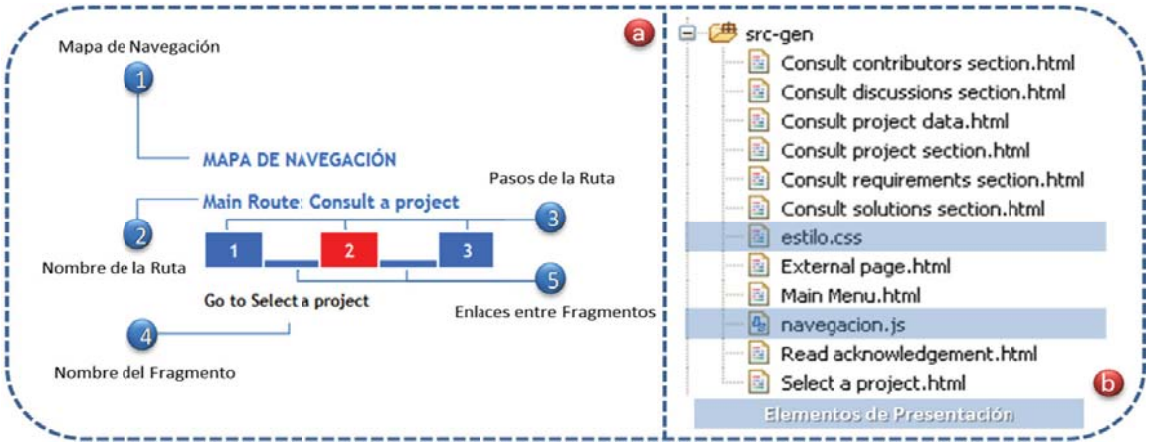

Figura 10 – Caso de Estudio: (a) Mapa de navegación (b) Ficheros generados

4. Conclusiones y Trabajos Futuros

En este artículo hemos presentado M2DAT-HYMO, el módulo de M2DAT que da soporte al Método del Modelado del Hipertexto de MIDAS (HM3). En esencia, M2DAT-HYMO implementa varios DSLs utilizando EMF como marco de metamodelado y GMF para la construcción de diagramadores, mientras que la automatización del proceso de desarrollo descansa en la implementación de varias transformaciones modelo a modelo y modelo a texto. Para la implementación de las primeras se ha utilizado el lenguaje ATL y las segundas se han desarrollado con Xpand.

A diferencia de otras herramientas, como ArgoUWE o HyperDE, en las que los modelos generados por algunas transformaciones deben ser refinados manualmente, M2DAT-HYMO permite añadir información adicional utilizando modelos de weaving para controlar la ejecución de la transformación. Además, los SIWs generados por M2DAT-HYMO incluyen rutas de navegación que ayudan a mejorar la usabilidad y navegabilidad del sistema.

Por otro lado, la Ingeniería Web está en constante avance y desarrollo y esto ha hecho que sus principios básicos (navegabilidad, usabilidad, etc.) hayan evolucionado de igual manera. Así, durante los últimos años ha surgido un nuevo tipo de aplicaciones denominadas RIAs (*Rich Internet Application*) (Fraternali et al., 2010) que suponen un salto cualitativo para los SIWs. Aquellos SIWs que hasta ahora sólo mostraban datos, se convierten en lugares que incluyen objetos multimedia y donde aumenta el nivel de interactividad con los usuarios, lo que posibilita que éstos puedan llevar a cabo tareas mucho más complejas. Dado que la demandad de este tipo de aplicaciones crece cada día, las metodologías de desarrollo Web se encuentran inmersas en un proceso de adaptación a estas nuevas necesidades. Así, una de las líneas futuras de este trabajo pasa por adaptar HM3 para soportar el desarrollo de este nuevo tipo de aplicaciones. Otras metodologías como UWE (Koch et al., 2009), OOHDM (Rossi et al, 2008) o

WebML (Comai & Toffetti Carughi, 2007) ya han empezado con este proceso de adaptación añadiendo nuevos modelos o combinando los suyos con otros ya existentes.

Por otro lado, en la sección 3.5 se mencionaba que HM3 no incluye por ahora un modelo de presentación. Por lo tanto, otra línea futura pasa por añadir un modelo de presentación que soporte la especificación de la interfaz gráfica del SIW. De hecho, la adaptación de HM3 para soportar la generación de RIAs hace imprescindible la inclusión de un modelo de presentación, ya que una de las ideas principales de las RIAs es ofrecer una interfaz muy compleja y estructurada en una sola pantalla, siguiendo el paradigma de la ventana única (*single window paradigm*). En esta misma dirección viene trabajando otras propuestas para la Ingeniería Web. En particular, las metodologías OOHDM y WebML, que ya disponían de modelo de presentación, lo han adaptado a los nuevos escenarios de las RIAs (Comai & Toffetti Carughi, 2007; Rossi et al., 2008) o han adoptado los modelos de presentación de otras metodologías; como RUX que clasifica la construcción de interfaces en interfaz abstracta, concreta y final (Preciado et al., 2007).

Agradecimientos

Este trabajo se ha llevado a cabo en el marco de los proyectos MODEL CAOS (Ref. TIN2008-03582) y Agreement Technologies (CONSOLIDER CSD2007-0022) financiado por el Ministerio de Ciencia y Tecnología de España.

Referencias

Budinsky, F., Merks, E., & Steinber, D. (2008). *Eclipse Modelling Framework 2.0* (Segunda ed.). Addison-Wesley Professional.

Cáceres, P., De Castro, V., Vara, J., & Marcos, E. (2006). Model transformations for hypertext modelling on web information systems. *SAC*, 1232-1239.

Comai, S., & Toffetti Carughi, G. (2007). A Behavioral Model for Rich Internet Applications. *Proceedings of the 7th International Conference Web Engineering (ICWE'07), LNCS 4607)*, (págs. 364-369).

De Castro, V. (2007). *Aproximación MDA para el Desarrollo Orientado a Servicios de sistemas de Información Web: del Modelo de Negocio al Modelo de Composición de Servicios Web*. Madrid: Universidad Rey Juan Carlos.

De Castro, V., Marcos, E., & Wieringa, R. (Junio de 2009). Towards a Service-oriented MDA-Based Approach to the Alignment of Business Processes with it Systems: From the Business Model to a Web Service Composition Model. *International Journal on Cooperative Systems, 18*(2), 225-260.

Didonet Del Fabro, M., Bézivin, J., & P., V. (2006). Weaving *Models with the Eclipse AMW plugin*. Esslingen, Alemania: Eclipse Modeling Symposium.

Efftinge, S. (22 de Julio de 2007). *Xtend Language Reference*, 4.1. Obtenido de http://www.eclipse.org/gmt/oaw/doc/4.1/r25_extendReference.pdf

Fraternali, P., Comai, S., Bozzon, A., & Toffetti, G. (2010). Engineering Rich Internet Applications with a Model-Driven Approach. ACM Transactions on the Web, Vol. 4, No. 2, Article 7.

Jouault, F., & Piers, W. (2009). *ATL User Guide*. Obtenido de http://wiki.eclipse.org/ATL/User_Guide

Klatt, B. (2007). A Closer Look at the model2text Transformation Language. *Chair for Software Design and Quality (SDQ)*.

Knapp, A., Koch, N., Moser, F., & Zhang, G. (2003). ArgoUWE: A Case Tool for Web Applications. *First International Workshop on Engineering Methods to Support Information Systems Evolution (EMSISE'03)*. Geneva, Switzerland.

Koch, N., Pigerl, M., Zhang, G., & Morozova, T. (2009). Patterns for the Model-based Development of RIAs. *Proceedings of the 9th International Conference Web Engineering (ICWE'09), LNCS 5648*, (págs. 283-291).

Lowe, D. (2003). Web System Requirements: An Overview. *Requirements Engineering Journal, 8*, 102-113.

Marcos, E., Acuña, C., & Cuesta, C. (2006). Integrating Software Architecture into a MDA Framework. *Proc. of 3rd European Workshop on Software Architecture* (págs. 27-143). France, Nantes: Springer Verlag, LNCS 4344.

Mernik, M., Heering, J., & Sloane, A. (2005). When and How to Develop Domain-Specific Languages. *ACM Computing Surveys, 37*(4), 316-344.

Miller, J., & Mukerji, J. (1 de Junio de 2003). *MDA Guide*, 1.0.1. Obtenido de http://www.omg.com/mda

Nielsen, J., & Loranger, H. (2006). Prioritizing Web Usability. *New Riders Publishing*.

Nunes, D., & Schwabe, D. (2006). Rapid Prototyping of Web Applications Combining Domain Specific Languages and Model Driven Design. *6th International Conference on Web Engineering (ICWE'06)*. Palo Alto, California, USA.

Oldvik, J., Neple, T., Gronmo, R., Aagedal, J., & Berre, A.-J. (2005). Toward Standardized Model to Text Transformations In Model Driven Architecture. *Foundations and Applications*, 239-253.

OMG. (2002). *MOF 2.0 Query/View/Transformations RFP*. Obtenido de OMG document ad/2002-04-10

Palmer, J. (2002). Web Site Usability, Design and Performance Metrics. *Information Systems Research, XIII*(2), 151-167.

Powell, A. (2004). *Generate Code with Eclipse's Java Emitter Templates*. IBM Developer Works.

Preciado, J.C., Linaje, M., Comai, S., & Sánchez-Figueroa, F. (2007). Designing Rich Internet Applications with Web Engineering Methodologies. *International Symposium on Web Site Evolution*, 23-30.

Rossi, G., Urbieta, M., Ginzburg, J., Distante, D., & Garrido, A. (2008). Refactoring to Rich Internet Applications. A Model-Driven Approach. *Proceedings of the 8th International Conference Web Engineering (ICWE'08)*, (págs. 1-12).

Schmidt, D. (2006). Model-Driven Engineering. *IEEE Computer*(39).

Selic, B. (2001). MDA Manifestations. *Upgrade. The European Journal for the Informatics Professional, IX*(2), 12-16.

Valverde, F., Valderas, P., Fons, J., & Pastor, O. (2007). A MDA-Based Environment for Web Applications Development: From Conceptual Models to Code. *6th International Workshop on Web-Oriented Software Technologies (IWWOST)*. Como, Italy.

Vara, J.M. (2009). M2DAT: a Technical Solution for Model-Driven Development of Web Information Systems. Madrid: Universidad Rey Juan Carlos.