Capítulo 16. Arquitectura Dirigida por Modelos (MDA).

Título de la nota 23/04/2005

La MDA se basa en separar la forma en que funciona un sistema de la forma en que este sistema usa las capacidades de la plataforma sobre la que se va a ejecutar.

MDA proporciona un enfoque para:

1) Especificar sistemas independientemente de la plataforma sobre la que se ejecutarán.
2) Especificar plataformas.
3) Elegir una plataforma concreta para el sistema.
4) Transformar la especificación del sistema independiente de la plataforma en una dependiente de ésta.

Este enfoque, además, paibilita el desarrollo de herramientas que le den soporte.

Los tres objetivos principales de la MDA son la interoperabilidad, la portabilidad y la reutilización mediante el adecuado diseño arquitectónico de los sistemas.

2. Definiciones.

2.1. Sistema.

2.2. Modelo.

Un modelo es una especificación formal (en el sentido de que utiliza un lenguaje bien definido) de las funciones, estructura y comportamiento de un sistema.

Habitualmente, el modelo es una combinación de dibujos y texto.

2.3. Dirigido por modelos ("Model-driven").

En MDA, "Dirigido por modelos" denota un mecanismo para comprender, diseñar, construir, distribuir ("deployment"), utilizar, mantener y modificar un sistema.

2.4. Arquitectura.

La arquitectura de un sistema es una especificación de las partes que lo componen, de sus conectores y de las reglas que describen las interacciones del sistema usando dichos conectores.

MDA utiliza diversos tipos de modelos y las relaciones entre éstos.

2.5. Punto de vista ("Viewpoint").

Un punto de vista es una forma de abstracción en la que se utilizan sólo un subconjunto de elementos arquitectónicos, de forma que uno puede centrarse exclusivamente en los aspectos del sistema que le interesan.

[En sistemas desarrollados usando el P.U. y UML, el diagrama de casos de uso podría ser la "vista funcional", el de clases la "vista de diseño", etc.].

MDA especifica tres puntos de vista:

- El independiente de la computación.
- El independiente de la plataforma.
- El dependiente de la plataforma (lo específico de la plataforma).

2.6 Vista ("View").

Una vista es el modelo utilizado para representar un sistema desde cierto punto de vista.

2.7. Plataforma ("Platform").

Conjunto de subsistemas y tecnologías que permiten un conjunto coherente de funcionalidades mediante interfaces y patrones de utilización delicadamente especificados. Cualquier aplicación soportada por la plataforma puede utilizar tales elementos sin indagar ni preocuparse por los detalles de implementación de la plataforma.

2.8. Aplicación.

En MDA, una aplicación es una funcionalidad que está siendo desarrollada.

Un sistema se descibe en forma de una o más aplicaciones soportadas por una o más plataformas.

3. Los puntos de vista en MDA

3.1. Modelo independiente de la computación (CIM: "Computation Independent Model").

Recordando la definición de "Modelo" que hemos dado más arriba, un CIM es una representación del sistema que se hace sin mostrar los detalles de su estructura. También se le llama "modelo de dominio", y suele utilizarse el lenguaje de las personas para las que se va a construir la aplicación final. De hecho, es el cliente el principal usuario del CIM.

3.2. Modelo independiente de la plataforma (PIM: "Platform Independent Model").

El PIM representa una vista del sistema especificada con un cierto grado de independencia de la plataforma final, de modo que tal modelo sería válido para plataformas de características similares.

3.3. Modelo específico de la plataforma (PSM: "Platform Specific Model").

Un PSM es un modelo del sistema que combina el PIM con los detalles que indican la forma en que el sistema usa dicha plataforma.

3.4. Modelo de plataforma ("Platform Model").

El PM describe las partes que conforman una plataforma y los servicios que suministra. También debe incluir la forma en que las aplicaciones deben usarla.

3.5. Transformación de modelos.

Es el proceso de transformar un modelo en otro, ambos representando el mismo sistema.

Normalmente, para transformar un PIM en un PSM, el proceso de transformación necesita algún tipo de información adicional, que se obtiene del PM. Además, suelen distinguirse tres tipos de transformaciones:

(a) Transformación de modelos en función del tipo.
(b) " por marcas.
(c) " mixtas.

3.5.1. Transformación de modelos en función del tipo.

En estas transformaciones, se describe cómo transformar cualquier PIM construido usando tipos especificados en el lenguaje del PIM a un PSM que usa tipos propios de tal PSM.

En estas transformaciones, se especifican reglas de transformación, que se describen en función de los valores de los miembros de los tipos del PIM.

P. ej.: usamos clases UML para representar el PIM. Cada clase tiene un atributo llamado "compartible", que vale true si la instancia podrá ser observada por múltiples observadores o no. Puede crearse una regla que indique que, al transformar una clase que se ha marcado como "compartible", se obtenga un EJB, y que se obtenga una clase Java estándar en caso contrario.

3.5.2. Transformación por marcas

En este caso, se identifican los elementos del PIM que deben ser transformados y cómo deben serlo.

Las reglas de transformación son establecidas por el arquitecto de Software y deben tener sentido: p. ej., puede "marcarse" el extremo de una asociación dirigida como "navegable por rmi", pero probablemente no tenga sentido decir que es una "entidad".

Las reglas de transformación de elementos del modelo se indican muchas veces en forma de "marcas", que son conceptos propios de la plataforma final pero que anotan (o "marcan") los elementos del PIM (aunque sin llegar a ser parte de éste).

3.5.3. Transformaciones mixtas.

La mayoría de las transformaciones combinan la transformación basada en tipos y la transformación de instancias.

Las transformaciones basadas en tipos son siempre deterministas, en el sentido de que no se pueden establecer reglas de transformación específicas, ya que todas se encuentran predeterminadas. Por ello, combinar la transformación basada en tipos y la basada en marcas es la que ofrece mayor poder de transformación.

En los procesos de transformación se utilizan plantillas ("templates"), que son elementos especiales de modelado que se utilizan para transformar elementos específicos.

Una vez se disponga del PIM adecuadamente marcado, la obtención del PSM se puede obtener manualmente, semiautomáticamente o de forma totalmente automática. En ocasiones, se obtiene directamente código ejecutable sin pasar por el PSM, aunque es recomendable producir uno para favorecer la comprensión del código.

Igualmente, puede generarse un "mapa de la transformación", que guarda información acerca de la correspondencia entre los elementos del PIM y del PSM.

3.5.4. Otras posibilidades de transformación.

Además de los tres enfoques anteriores, es posible mezclar dos o más PIM para producir un único PSM; suministrar al proceso distinta del conjunto de marcas para guiar la transformación (p. ej., información sobre el estilo arquitectónico

de la aplicación final)

3.6. Requisitos para la transformación automática.

En ocasiones, el PIM contiene toda la información necesaria para la implementación, sin que sea necesaria la adición de marcas para generar el código. Esto ocurre, p. ej., en el desarrollo de aplicaciones basadas en componentes, en las que el estilo arquitectónico es el mismo para una variedad de sistemas muy grande, y en las que el propio middleware ofrece ya un conjunto amplio de funcionalidades.

En contextos como el del ejemplo, puede ocurrir que el desarrollador ni siquiera necesite ver ni manipular el PSM. Para que esto sea posible, es claro que alguien ha debido describir, en un formato reutilizable:

(1) Un estilo arquitectónico
(2) Un sistema de tipos para el PIM, que permita la transformación desde éste hacia, quizás, varias plataformas.

(3) Un conjunto de herramientas que permitan a los desarrolladores trabajar con el modelo.

(4) Un mecanismo de transformación para cada plataforma de destino.

4. Otras aplicaciones del MDA.

4.1. Modelos multiplataforma.

El PIM se puede anotar con marcas de plataformas diversas para obtener diversos PSM.

4.2 Sistemas federados.

El PIM puede constar de varias partes, pudiendo obtenerse de cada una un PSM diferente.

5. El UML como lenguaje ejecutable (xUML: "eXecutable UML").

UML es un lenguaje para el modelado de sistemas software estandarizado por el OMG, consorcio de empresas que mantiene también el control sobre los estándares de MDA.

UML está ampliamente aceptado y es muy adecuado para representar la estructura y el comportamiento de sistemas. Tiene, sin embargo, algunos problemas importantes, como la ambigüedad. Los modelos en UML pueden ser anotados con restricciones OCL para darles un toque más formal y mejorar sus características de generación de código.