Entorno para Coreografiar Movimientos en un Robot Humanoide

Juan F. Inglés-Romero, Cristina Vicente-Chicote, Diego Alonso
Escuela Técnica Superior de Ingeniería de Telecomunicación
División de Sistemas e Ingeniería Electrónica (DSIE)
Antiguo Cuartel de Antigones. Plaza del Hospital, Nº 1, 30202 Cartagena (Murcia)
Teléfono: 968326448
E-mail: {juanfran.ingles, cristina.vicente, diego.alonso}@upct.es

Resumen. *En los últimos años, el Desarrollo Software Dirigido por Modelos ha ido ganando en importancia y, actualmente, es considerado uno de los enfoques más prometedores en el ámbito de la Ingeniería del Software. En este artículo, se presenta un ejemplo práctico de aplicación de este nuevo enfoque al dominio de la robótica, centrado en el desarrollo de un entorno para modelar coreografías de movimientos para el robot humanoide Robonova. Este entorno permite a los usuarios de esta plataforma robótica modelar gráficamente y validar formalmente las secuencias de movimientos del robot (coreografías), así como generar automáticamente la implementación asociada a cada coreografía en el lenguaje de programación del robot, denominado RoboBASIC.*

1 Introducción

El **DSDM** (*Desarrollo de Software Dirigido por Modelos*) [1] permite elevar el nivel de abstracción con el que se diseña el software y ofrece mecanismos para automatizar, en gran medida, el proceso de generación de código. El DSDM comprende un conjunto de técnicas y herramientas que permiten modelar formalmente los sistemas que se quieren desarrollar para, posteriormente, aplicando una serie de transformaciones automáticas, obtener el código final de las aplicaciones.

El DSDM gira entorno a la definición y el uso sistemático de modelos a lo largo de todo el ciclo de vida del software. Los modelos proporcionan una simplificación de un sistema real, construida con un determinado fin. Estos modelos han de ser capaces de responder a las preguntas que se les formulen, como si se tratara del sistema real. Las respuestas que proporcionan los modelos deben ser exactamente las mismas que si respondiera el sistema real, a condición de que las preguntas se formulen en los mismos términos en los que se definió el modelo. Para definir cualquier modelo es necesario contar con un lenguaje de modelado. Los meta-modelos definen formalmente la sintaxis abstracta de los lenguajes de modelado, recogiendo sus conceptos (palabras del lenguaje), así como las reglas que indican cómo se pueden combinar dichos conceptos para formar modelos válidos [1].

El DSDM se ha aplicado de forma exitosa en muchos dominios, algunos relativamente cercanos a la robótica como el de los sistemas empotrados [2] o las redes de sensores [3]. Sin embargo, las referencias al DSDM en el ámbito de la robótica son recientes y aún escasas [4-6], aunque algunos autores identifican este nuevo paradigma como una de las claves para solventar la "falta crónica de normalización, interoperabilidad y reutilización del software en robótica" [5].

El entorno de modelado que se presenta en este trabajo, ofrece una primera aproximación al desarrollo de software para robótica utilizando un enfoque dirigido por modelos. En su versión actual, este nuevo entorno de modelado sólo da soporte al robot humanoide de bajo coste y utilizado fundamentalmente con fines docentes, conocido como Robonova (http://www.robonova.com), si bien se están barajando posibles extensiones para dar soporte a otras plataformas robóticas.

2 Descripción del entorno

A continuación se describen las dos herramientas que conforman el entorno de modelado propuesto (ver Fig. 1), ambas desarrolladas utilizando las facilidades para DSDM que ofrece la plataforma Eclipse (www.eclipse.org). Así, la Sección 2.1 describe la herramienta gráfica de modelado y validación de coreografías, implementada utilizando **GMF** (*Graphical Modelling Framework*) y la Sección 2.2, la transformación **M2T** (*Model-to-Text*), realizada con **JET** (*Java Emitter Templates*), para generar el código RoboBASIC (www.robobasic.com) a partir de los modelos anteriores.

2.1 Editor gráfico de coreografías

El entorno de modelado que se describe en este artículo incluye una herramienta gráfica para modelar, en un entorno amigable e intuitivo, coreografías de movimientos para robots Robonova. Así, cualquier usuario puede especificar, de manera totalmente 'visual', las coreografías del robot sin necesidad de conocer su lenguaje de programación.

El meta-modelo sobre el que se ha construido esta herramienta (ver Fig. 2), incluye los conceptos necesarios para modelar las coreografías de movimientos de los robots Robonova.

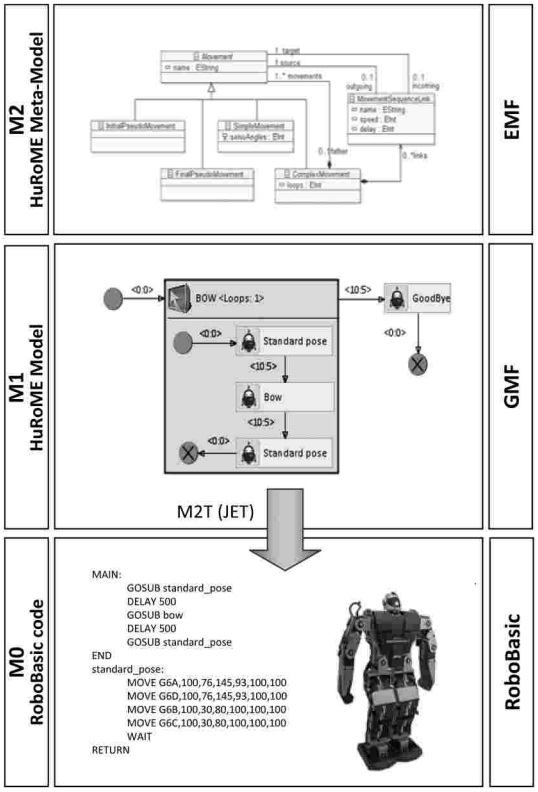

Fig. 1 Etapas del proceso de modelado.

Este meta-modelo recoge los siguientes conceptos de modelado: (1) *SimpleMovement*, modela los cambios de postura logrados mediante el accionamiento de uno o más de sus actuadores mecánicos; (2) *ComplexMovement*, modela la composición jerárquica de movimientos; (3) *PseudoMovement*, modela los puntos de control que establecen el inicio y el final de cada secuencia; y (4) *MovementSequenceLink*, enlaza parejas de movimientos indicando el orden en que éstos se ejecutan. Entre los atributos asociados a cada uno de estos conceptos, encontramos los valores angulares de las articulaciones en cada *SimpleMovement*, el número de veces que se repite cada *ComplexMovement*, o el retardo y la velocidad asociada a los *MovementSequenceLink*.

La herramienta de modelado desarrollada como parte de este trabajo, proporciona una sintaxis concreta (en este caso gráfica), para cada uno de los conceptos del meta-modelo (sintaxis abstracta). En la Fig. 3 se muestra el editor gráfico de modelos desarrollado para permitir la creación de coreografías.

Este editor consta de tres partes: (1) una paleta de herramientas (panel derecho) desde la que el usuario puede seleccionar los conceptos que desea incorporar a sus modelos, (2) un área de trabajo (panel central) donde podrá modelar las coreografías de movimientos del robot, y (3) una vista de propiedades (panel inferior) para añadir y completar la información asociada a los distintos elementos del modelo (por ejemplo, el valor angular de cada uno de los servomotores del robot en cada *SimpleMovement*).

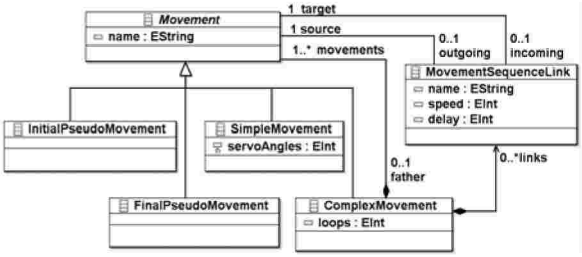

Fig. 2. Meta-modelo sobre el que se ha desarrollado el entorno de modelado.

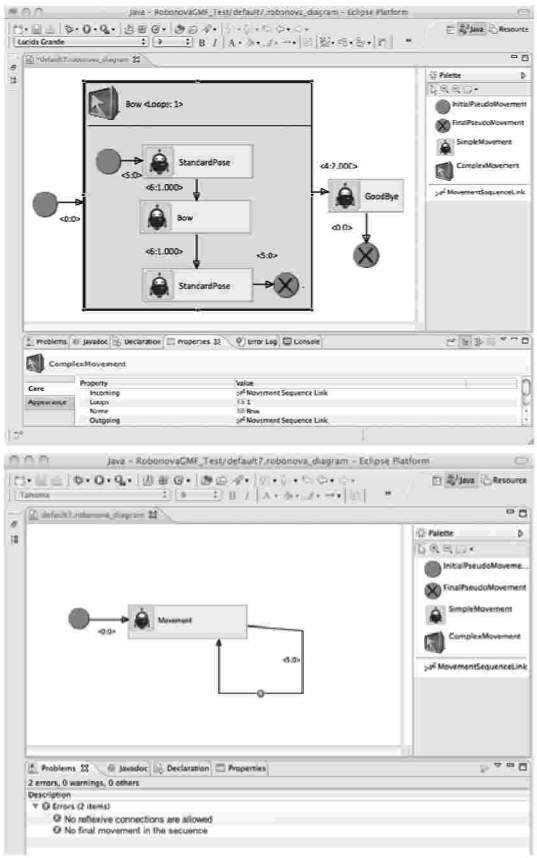

Fig. 3. *Arriba*: Ejemplo de coreografía especificada con la herramienta gráfica de modelado. *Abajo*: Errores de validación detectados por la herramienta.

La herramienta gráfica de modelado incorpora adicionalmente facilidades de validación, que permiten comprobar formalmente la corrección sintáctica de los modelos, comprobando si éstos son conformes al meta-modelo y a una serie de restricciones adicionales, escritas en lenguaje **OCL** (*Object Constraint Language*) y añadidas al editor. Entre las restricciones incluidas en este caso, cabe destacar las siguientes: (1) no se permiten las conexiones reflexivas (conectar un movimiento consigo mismo); (2) sólo se pueden enlazar movimientos del mismo nivel de composición; (3) el elemento inicial de una secuencia tiene sólo un link de salida y el final sólo un link de entrada; (4) todos los movimientos, simples o compuestos, tienen un único link de entrada y otro de salida, salvo la excepción descrita en 3.

Cuando se detecta una violación de estas reglas o de las recogidas explícitamente en el meta-modelo, los elementos afectados son marcados con un círculo rojo y una cruz (ver Fig. 3, abajo).

Sólo cuando los modelos están totalmente libres de errores de validación, puede aplicárseles la transformación que se describe a continuación para generar el código correspondiente para el robot.

2.2 Generación de código RoboBASIC

El objetivo de esta herramienta es convertir, mediante una transformación automática, los modelos diseñados y validados con el editor gráfico descrito en la Sección 2.1, en código RoboBASIC que pueda ser ejecutado en un robot Robonova. La Tabla 1 refleja las estructuras del lenguaje RoboBASIC a las que se traducen los distintos conceptos definidos en el meta-modelo descrito en la sección anterior.

El procedimiento de transformación lo dirige el motor de ejecución de JET. Éste recorre el modelo especificando qué plantilla de código se debe invocar para cada tipo de elemento encontrado. Así, según se indica en la Tabla 1, las instancias de la clase *ComplexMovement* son traducidas en bucles que se repiten el número de veces indicado por el atributo *loops*. El código que se incluirá en cada bucle vendrá determinado por el resultado de procesar los elementos contenidos en cada *ComplexMovement*. Por otro lado, los elementos *SimpleMovement* se transforman en una o más sentencias de movimiento simultáneo de los servomotores, delimitadas con la clave WAIT. De este modo, el robot transiciona a una pose determinada por el valor de la propiedad *servoAngles*. Por último, las instancias de la clase *MovementSequenceLink* se traducen a sentencias que controlan la velocidad y el retardo de los movimientos, de acuerdo a los valores establecidos en las propiedades *speed* y *delay*.

Tabla 1. Correspondencia entre conceptos del meta-modelo y estructuras RoboBASIC.

Conceptos del dominio	Código RoboBASIC
ComplexMovement	DIM I AS INTEGER FOR I=0 TO *loops* *(Código elementos contenidos)* NEXT
SimpleMovement	MOVE *servoAngles* … WAIT
MovementSequenceLink	SPEED *Speedy* DELAY *delay*

3 Conclusiones y trabajos futuros

Este artículo demuestra, mediante un ejemplo sencillo, algunos de los beneficios que puede aportar el uso de un enfoque de DSDM a la robótica.

El entorno desarrollado permite a los diseñadores modelar gráficamente y validar formalmente coreografías de movimientos para los robots Robonova. Estas coreografías pueden transformarse automáticamente en el código roboBASIC equivalente para ser ejecutadas por el robot. Todas las herramientas desarrolladas como parte de este trabajo han sido implementadas usando algunas de las herramientas para DSDM que ofrece la plataforma Eclipse: EMF, GMF, OCL y JET.

Como trabajos futuros se baraja la extensión de las herramientas para permitir la creación de librerías de movimientos reutilizables, así como la implementación de una transformación **T2M** (Text-to-Model), que permita la generación de modelos de coreografías a partir de cualquiera de los muchos ficheros de código ya existentes.

Agradecimientos

Este trabajo ha sido parcialmente financiado por el Proyecto CICYT EXPLORE (TIN2009-08572).

Referencias

[1] T. Stahl, M. Voelter, and K. Czarnecki, Model-Driven Software Development: Technology, Engineering, Management, ed. Wiley, 2006, ISBN: 978-0-470-02570-3.

[2] QUASIMODO: Quantitative system properties in model-driven design of embedded systems, http://www.quasimodo.aau.dk/

[3] C. Vicente-Chicote, F. Losilla, B. Álvarez, A. Iborra, P. Sánchez, Applying MDE to the Development of Flexible and Reusable Wireless Sensor Networks, Int'l Journal of Cooperative Information Systems, 16(3/4), 393—412, 2007.

[4] BRICS: Best Practice in Robotics, http://www.best-of-robotics.org/

[5] H. Bruyninckx, Robotics software: the future should be open. IEEE Robotics & Automation Magazine, 15(1), 9—11, 2008.

[6] D. Alonso, et al., V3CMM: a 3-View Component Meta-Model for Model-Driven Robotic Software Development, Journal of Software Engineering for Robotics, 1 (1), 3—17, 2010.