HuRoME: Entorno para Modelado de Coreografías y Modernización de Código para un Robot Humanoide

Juan F. Inglés-Romero
División de Sistemas e Ingeniería Electrónica (DSIE)
Universidad Politécnica de Cartagena
30.202 Cartagena, España
juanfran.ingles@upct.es

Cristina Vicente-Chicote
División de Sistemas e Ingeniería Electrónica (DSIE)
Universidad Politécnica de Cartagena
30.202 Cartagena, España
cristina.vicente@upct.es

Diego Alonso
División de Sistemas e Ingeniería Electrónica (DSIE)
Universidad Politécnica de Cartagena
30.202 Cartagena, España
diego.alonso@upct.es

Resumen

El entorno de modelado **HuRoME** (*Humanoid Robot Modeling Environment*) integra un conjunto de herramientas diseñadas para facilitar el modelado de coreografías y la modernización del software existente para el robot humanoide Robonova (www.robonova.com).

HuRoME permite a los numerosos usuarios de Robonova: (1) modelar gráficamente y validar formalmente las secuencias de movimientos del robot (coreografías), (2) generar automáticamente la implementación asociada a cada coreografía en el lenguaje RoboBASIC, y (3) modernizar y reutilizar el software ya existente, permitiendo la obtención de los modelos equivalentes a cualquier programa RoboBASIC.

1. Introducción

El Desarrollo de Software Dirigido por Modelos (DSDM) [5] 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. Este paradigma gira entorno a la definición y el uso sistemático de modelos y transformaciones de modelos a lo largo de todo el ciclo de vida del software.

El DSDM se ha aplicado de forma exitosa en muchos dominios, algunos relativamente cercanos a la robótica como los sistemas empotrados [4] o las redes de sensores [6]. Sin embargo, las referencias al DSDM en el ámbito de la robótica son recientes y aún escasas [1-2], 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*" [3].

El entorno **HuRoME**, que se presenta en este trabajo, ofrece una primera aproximación al desarrollo de software para robótica utilizando un enfoque dirigido por modelos. Como se detalla en las siguientes secciones, **HuRoME** no sólo facilita la generación automática de código a partir de modelos, sino también el proceso inverso, permitiendo la obtención de modelos a partir de programas ya existentes (*legacy code*), facilitando así su tratamiento (depuración, extensión, reutilización, análisis, simulación, etc.) con un nivel de abstracción mayor que el que proporciona el código fuente. En su versión actual, **HuRoME** sólo da soporte a la plataforma Robonova (robot humanoide de bajo coste utilizando fundamentalmente con fines docentes), si bien se están barajando posibles extensiones para dar soporte a otras plataformas robóticas.

2. Descripción de HuRoME

A continuación se describen las tres herramientas que conforman **HuRoME** (ver Figura 1), todas ellas desarrolladas utilizando las facilidades para DSDM que ofrece la plataforma Eclipse. En primer lugar, la Sección 2.1 describe la herramienta gráfica de modelado y validación de coreografías, implementada con GMF (*Graphical Modeling Framework*). A continuación, la Sección 2.2, describe la transformación modelo-a-texto (M2T), implementada con JET (*Java Emitter Templates*), para generar código RoboBASIC a partir de los modelos anteriores. Por último, la Sección 2.3, describe la transformación texto-a-modelo (T2M), implementada con xText y ATL (*ATLAS transformation language*), para obtener modelos a partir de código RoboBASIC.

Actas de las JISBD 2010, pp. 251-254, ISBN: 978-84-92812-51-6

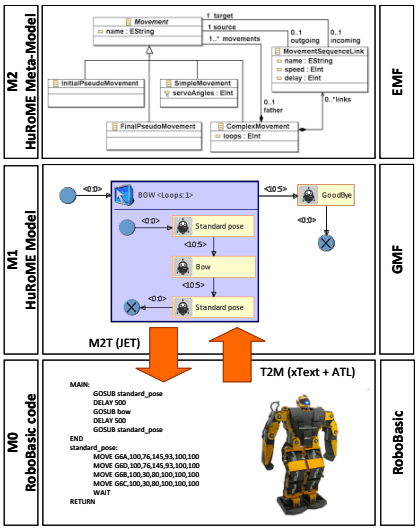

Figura 1. Herramientas integradas en **HuRoME**.

2.1. Editor gráfico para modelado y validación

HuRoME ofrece una herramienta gráfica de modelado que permite diseñar, 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 Figura 2), incluye los conceptos necesarios para modelar las coreografías de movimientos de los robots Rovonova, en particular: (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*.

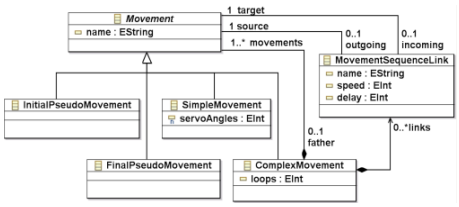

Figura 2. Meta-Modelo **HuRoME**.

La herramienta de modelado proporciona una sintaxis concreta (en este caso gráfica), para cada uno de los conceptos del meta-modelo (sintaxis abstracta). En la Figura 3 se muestra el editor gráfico de modelos desarrollado como parte de **HuRoME.** Esta 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*).

La herramienta de modelado, implementada como parte de **HuRoME,** incorpora 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 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 Figura 3). 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.

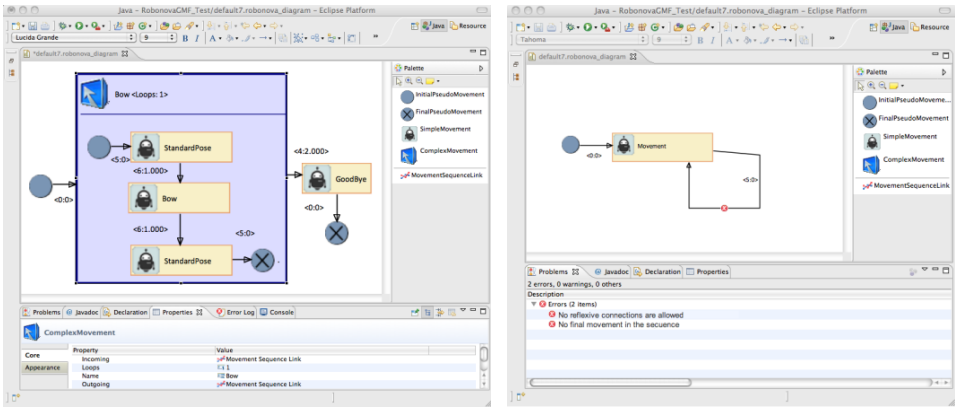

Figura 3. *Izquierda*: Coreografía de movimientos especificada con la herramienta de modelado de **HuRoME**. *Derecha*: Errores de validación detectados por la herramienta.

2.2. Generación de código RoboBASIC

El objetivo de esta herramienta es convertir, mediante una transformación automática modelo-a-texto (M2T), 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.

TRANSFORMACIÓN MODELO A TEXTO

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 *speed* DELAY *delay*

Tabla 1. Correspondencia entre los conceptos del meta-modelo y las estructuras de código RoboBASIC.

Así pues, 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 movimientos simultáneos 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 con los valores establecidos en las propiedades *speed* y *delay*.

2.3. Generación de modelos a partir de código

Con el objetivo de soportar el proceso inverso al descrito en el apartado anterior, la versión actual de **HuRoME** incorpora una transformación texto-a-modelo (T2M) que permite obtener modelos de coreografías a partir de código RoboBASIC v1.0 ya existente (bien generado automáticamente por la herramienta anterior o codificado manualmente por un desarrollador). En una primera fase, partiendo del código RoboBASIC v1.0, se realiza una transformación T2M para obtener un modelo conforme al meta-modelo de xText. Asociado a esta fase se ha desarrollado un editor textual

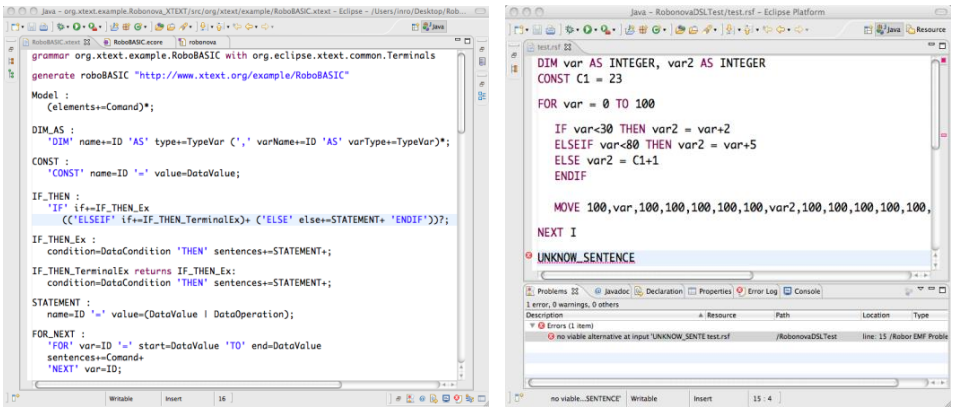

Figura 4. *Izda.*: Gramática RoboBASIC v1.0. *Dcha.*: Editor textual de código RoboBASIC con validador sintáctico.

para el lenguaje RoboBASIC, que permite el formateo léxico y la validación sintáctica del código (ver Figura 4). En una segunda fase, se ha implementado en ATL una transformación M2M con el fin de obtener modelos conformes al meta-modelo de **HuRoME** a partir de los modelos xText obtenidos en la etapa anterior.

3. Planificación de la Demostración

La demostración de la herramienta se llevará a cabo en tres partes: (1) diseñando un modelo gráfico sencillo de movimientos como el mostrado en el ejemplo de la Figura 2, (2) mostrando cómo **HuRoME** permite la detección de errores de modelado en los diseños, (3) a partir de un modelo correctamente validado, se generará el código RoboBASIC v1.0 correspondiente, que se compilará y ejecutará sobre un robot Robonova para que lleve a cabo la coreografía modelada y, por último, (4) a partir del código de cualquiera de los programas de ejemplo disponibles para el robot en múltiples páginas web, se generará el modelo correspondiente, que será modificado y vuelto a re-codificar automáticamente, mostrando así la sencillez y potencia de **HuRoME**.

Agradecimientos

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

Referencias

[1] 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.

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

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

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

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

[6] 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.