Aplicando Técnicas de Ingeniería de Software al Desarrollo de Sistemas Robóticos Educativos

Claudia Pons, Gonzalo Zabala and Gabriela Arévalo
CONICET, Facultad de Informática de la UNLP y Universidad Abierta Interamericana (UAI)
Buenos Aires, Argentina

RESUMEN

La inclusión del sistema educativo argentino al modelo de "una computadora portátil por niño" a través del programa "Conectar Igualdad", entre otros, junto con el sólido propósito tanto del Ministerio de Educación como del Ministerio de Ciencia y Tecnología de equipar a las escuelas con artículos tecnológicos concretos, tales como robots y adquisidores de datos, nos da una oportunidad única de desarrollar plataformas robóticas / automatizadas dirigidas a la educación. Mientras que los sistemas robóticos crecen hasta ser más y más complejos, la necesidad de formalizar su proceso de desarrollo de software crece también. Los enfoques tradicionales que se utilizan en el proceso de desarrollo de estos sistemas de software están alcanzando sus límites; las metodologías y el conjunto de herramientas no alcanzan para atender / las necesidades de tal complejo proceso de desarrollo de software. El objetivo general de este proyecto es la definición de un marco metodológico sustentado por un conjunto de herramientas tecnológicas capaces de lidiar con los requerimientos del proceso de desarrollo del software robótico. Un desafío muy importante es dar el paso desde "code-driven" a "model-driven" en el desarrollo de sistemas de software de robótica. La separación del conocimiento de robótica de la aplicación de tecnologías de ciclo corto es esencial para fomentar el reuso y el mantenimiento. El contexto en el cual serán aplicados los resultados comprende el campo de desarrollo de los sistemas robóticos en general, pero los remitiremos específicamente a los sistemas robóticos educativos que podrían ser potencialmente usados en las aulas argentinas.

Palabras clave: robótica, ingeniería de software, educación.

1. CONTEXTO

Este proyecto cuenta con la participación de profesores / investigadores del CONICET (Consejo Nacional de Investigaciones Científicas y Técnicas), de la Universidad Abierta Interamericana (UAI), de la Universidad Nacional de La Plata (UNLP) y de la Universidad de la República (Uruguay). Los investigadores que participan en este proyecto se complementan muy bien de acuerdo a sus áreas de especialización; contribuyen al conocimiento en las áreas de modelado de software, ingeniería de software y robótica.

2. INTRODUCCIÓN Y PLANTEO DEL PROBLEMA

Los sistemas robóticos (SR) juegan un papel cada vez mayor en la vida cotidiana. También la necesidad de sistemas automatizados en entornos industriales y educativos aumenta y se vuelve más exigente. Mientras que los sistemas robóticos crecen y cada vez son más complicados, la necesidad de formalizar su proceso de desarrollo de software crece también. Los enfoques tradicionales que se utilizan en el proceso de desarrollo de estos sistemas de software están alcanzando sus límites; las metodologías utilizadas actualmente y los conjuntos de herramientas no alcanzan para atender las necesidades de dicho proceso de desarrollo de software complejo.

La inclusión del sistema educativo argentino al modelo de "una computadora portátil por niño" a través del programa "Conectar Igualdad" [1], entre otros, junto con el sólido propósito tanto del Ministerio de Educación como del Ministerio de Ciencia y Tecnología de equipar las escuelas con artículos tecnológicos concretos, tales como robots y adquisidores de datos, nos da una oportunidad única de desarrollar plataformas robóticas / automatizadas dirigidas a la educación.

Está ampliamente aceptado que nuevos enfoques deben ser establecidos para satisfacer las necesidades del complejo proceso de desarrollo SR de hoy. En esta dirección, el desarrollo basado en componentes (CBD, por sus siglas en inglés Component-based development) [5], la Arquitectura Orientada a Servicios (SOA, por sus siglas en inglés Service Oriented Architecture) [12] [13], así como la Ingeniería de software dirigida por modelos (MDE, por sus siglas en inglés Model Driven software Engineering) [15] [16] y el Modelado Específico de Dominio (DSM, por sus siglas en inglés Domain-Specific Modeling) [14] se encuentran entre las principales y prometedoras tecnologías en el ámbito de SR. En estos días,

estamos realizando una investigación sobre el uso actual de estas técnicas modernas de ingeniería de software para el desarrollo de sistemas de software robótico y su nivel de automatización real. Un desafío muy importante es dar el paso desde "code-driven" a "model-driven" en el desarrollo de sistemas de software de robótica

3. OBJECTIVOS, HIPÓTESIS Y RESULTADOS ESPERADOS

El objetivo general de este proyecto tiene dos partes: Desde el punto de vista técnico queremos definir un marco metodológico con el apoyo del proceso MDD para hacer frente a las exigencias del proceso e desarrollo de software robótico, y desde el punto de vista educativo, queremos desarrollar el software y hardware para complementar las netbooks existentes distribuidas en las escuelas primarias de Argentina. El contexto en el cual los resultados serán aplicados comprende el campo de desarrollo de sistemas robóticos en general, pero los vamos a dirigir específicamente a los sistemas educativos robóticos que pueden ser potencialmente utilizados en las aulas argentinas. Esas plataformas robóticas educativas deben poseer una capacidad de adaptación muy dinámica, acompañando a la tasa de desarrollo de esas tecnologías y a las diferencias locales de cada una de las plataformas de hardware. Debido a este hecho, es esencial contar con potentes herramientas de ingeniería de software que facilitan la construcción de frameworks para hacer frente a tal complejidad y volatilidad tecnológica.

Basado en el contexto y los objetivos explicados anteriormente, vamos a trabajar bajo las siguientes hipótesis, en las que combinamos nuestro conocimiento de los aspectos técnicos de los sistemas robóticos y el impacto social que generan el uso de nuestros resultados en las netbooks distribuidas en el programa "Conectar-Igualdad".

- Es indispensable propiciar la aplicación de los principios de ingeniería para hacer frente a la complejidad de los sistemas de software robóticos, porque no podemos esperar un crecimiento significativo con sistemas hechos artesanalmente.
- Las- Interfaces y el comportamiento de los sistemas robóticos se deben definir en un nivel superior de abstracción para que puedan ser reutilizados con diferentes plataformas tecnológicas. Lograr una separación de los conocimientos de robótica de las tecnologías de ciclo corto es esencial para fomentar la reutilización y el mantenimiento.
- La aplicación de tecnologías existentes de ingeniería de software, tales como SOA, MDE, y CBD, para construir sistemas de software robóticos podría ahorrar una gran cantidad de tiempo y esfuerzo, además de favorecer la reutilización y el mantenimiento de dichos sistemas.
  Dichas técnicas de ingeniería de software podrán ser aplicadas a la construcción de software educativo robótico dirigido a la enseñanza de la tecnología y la ciencia en las escuelas argentinas, dando lugar a mejoras en el rendimiento educativo.

En este contexto, los objetivos específicos del proyecto son los siguientes:

- Resumir la evidencia existente sobre la aplicación de las tecnologías de ingeniería de software, tales como SOA, MDE y el CBD en el campo de desarrollo de sistemas robóticos;
- Identificar las carencias en la investigación actual con el fin de sugerir áreas para futuras investigaciones;
- Proporcionar un fundamento teórico con el fin de posicionar adecuadamente las nuevas actividades de investigación;
- Mejorar las técnicas actuales para ofrecer un avance en el estado del arte;
- Definir las líneas de base para una metodología abierta para el proceso de desarrollo de software robótico.
- Crear herramientas de apoyo al proceso de desarrollo de software robótico. Ejemplos de estas herramientas son: un lenguaje de modelado específico de dominio equipado con editores gráficos, facilidades de generación de código, integración con servicios web, editores de definición de componentes , etc.
- Usar los resultados para la construcción de sistemas robóticos educativos que se utilizarán en las aulas argentinas a bajo costo.
- Realizar una serie de experimentos para evaluar la eficacia y la viabilidad de la utilización de sistemas robóticos para mejorar el proceso educativo de los niños argentinos

4. LINEAS DE INVESTIGACIÓN Y RELEVANCIA DEL PROBLEMA

Los primeros proyectos que se centraban en la aplicación de las computadoras en el ámbito educativo tenía el objetivo de vincular los elementos tecnológicos concretos con las herramientas digitales ofrecidas por esa tecnología. Un ejemplo es el desarrollo de Lego Logo, donde se controla un robot físico por medio de comandos que se introducen en la computadora. Tales órdenes causar un efecto tanto en el robot concreto, así como en su representación virtual. Por varias razones, esta primera propuesta de mantener la presencia de material no-digital en las áreas de educación tecnológica fue desapareciendo con los años, en proporción inversa a la introducción de

computadoras en las escuelas. Así, la educación tecnológica se refiere únicamente al mundo digital, perdiendo la conexión con el material de tecnología concreto. Debido al hecho de que ciertas estructuras cognitivas no se desarrollan en ausencia de material concreto, esta desconexión muy probablemente podría originar una grave laguna en la educación de los niños. La situación se hace aun más difícil por el hecho de que en la actualidad los juegos habituales de los niños también se asocian sobre todo con el mundo virtual.

En los últimos años, la aparición de kits de robótica orientados a usuarios no expertos dio lugar al desarrollo de un número significativo de proyectos educativos utilizando robots. Estos proyectos se aplican los robots en diferentes niveles educativos, desde pre-escolar hasta educación superior, especialmente en las áreas de la física y tecnología. Los dispositivos que son construidos por los alumnos que utilizan este material ofrecen la posibilidad de nuevos experimentos científicos relacionados con los fenómenos cotidianos. La fuerte motivación que el uso de estos materiales genera en los estudiantes ha alentado a los docentes de otras disciplinas para desarrollar propuestas de utilización de robots en el campo de las artes, las ciencias sociales y otros. Estas nuevas propuestas pueden revertir los déficits cognitivos causados por la no utilización de material concreto en juegos diarios y la educación formal.

En este contexto, uno de los problemas que nos encontramos es que el hardware de los kits de robótica está en constante cambio, además su uso no es uniforme en las diferentes regiones e incluso en los niveles de educación ni desde el punto de vista técnico. Por lo tanto, las interfaces técnicas de estos robots deben ocultar estas diferencias para que los educadores no se vean forzados a cambiar su material educativo una y otra vez. Un ejemplo de estas interfaces es "*Physical Etoys*" [2], un proyecto en el que participamos y que propone una plataforma de enseñanza estándar para la programación de robots, independientemente de si están basados en Arduino, Lego, u otras tecnologías. *Physical Etoys* ha logrado muy buena aceptación en la comunidad académica en todo el mundo y proporciona un punto de partida prometedor para el desarrollo de una poderosa plataforma de ingeniería de software para el desarrollo de sistemas robóticos aplicables en las aulas argentinas.

Aunque la complejidad del software robótico es alta, en la mayoría de los casos su reutilización todavía está restringida al nivel de bibliotecas. En el nivel inferior, se han creado una multitud de bibliotecas para realizar tareas como cálculos matemáticos para cinemática, dinámica y visión de la máquina, como por ejemplo [3]. En vez de construir los sistemas a partir de bloques que provean servicios ya verificados, el proceso de construcción de software de un nuevo sistema robótico a menudo sigue siendo la re-implementación de la lógica para unir las diferentes bibliotecas. A menudo, la integración global está orientada a un middleware determinado y sus capacidades. Los middlewares se utilizan para ocultar la complejidad de la comunicación entre componentes, por ejemplo OpenRTM-AIST [4] es un middleware basado en CORBA para las plataformas de robots. Obviamente, esto no sólo es caro y consume enormes recursos sino que desaprovecha la madurez alcanzada por los procesos de desarrollo de software existentes.

En concreto, los sistemas de robótica tienen necesidades especiales, a menudo relacionadas con su naturaleza de tiempo real y las propiedades del ambiente. Este tipo especial de sistemas necesita más calidad que un sistema de propósito general y tiene que ser capaz de hacer frente al entorno físico incierto y dinámico en el que están inmersos. Atributos como la fiabilidad y la seguridad son una necesidad en este dominio.

En este contexto, es ampliamente aceptado que nuevos enfoques deben ser aplicados para satisfacer las necesidades de este proceso de desarrollo tan complejo. El Desarrollo Basado en Componentes (CBD), la Arquitectura Orientada a Servicios (SOA), así como la Ingeniería de software Dirigida por Modelos (MDE) y el Modelado de Dominio Específico (DSM) se encuentran entre las tecnologías más prometedoras en el ámbito de SRs.

En primer lugar, el paradigma CBD [5] establece que el desarrollo de aplicaciones debe lograrse mediante la vinculación de partes independientes, los componentes. Las interfaces de los componentes, basadas en estrictos patrones de interacción, separan la esfera de influencia y por lo tanto particionan la complejidad global. Esto se traduce en componentes débilmente acoplados que interactúan a través de servicios con contratos. Los componentes como unidades arquitectónicas permiten especificar con mucha precisión, utilizando el concepto de puerto, tanto los servicios prestados como los servicios requeridos por un determinado componente. Además definen una estrategia de composición basada en la noción de conectores. La tecnología de componentes ofrece altas tasas de reutilización y facilidad de uso, pero poca flexibilidad con respecto a la plataforma de implementación. Los componentes actuales están relacionados con C / C + + y Linux (por ejemplo, Microsoft Robotics Developer Studio [6], EASYLAB [7], Player/Stage Project[10]), aunque algunos alcanzan más independencia, gracias a la utilización de alguno middleware (por ejemplo, Smart Software Component Model [11], Orocos [3] Orca [8] CLARAty [9]).

En segundo lugar, necesitamos una manera de definir las interfaces y comportamiento a un nivel más alto de abstracción para que puedan ser utilizados en sistemas con diferentes plataformas. Esto originó la idea de componentes abstractos, es decir, independientes de la plataforma de implementación. De este modo, para superar los problemas actuales resulta imperioso migrar de diseños basados en el código hacia diseños basados en los modelos. Una descripción basada en el modelo es un medio adecuado para expresar los contratos de las interfaces de los componentes, verificar el comportamiento general de los sistemas integrados y finalmente obtener automáticamente el software ejecutable. En lugar de construir herramientas de soporte para cada framework a partir de cero, ahora la estrategia consiste en expresar los modelos necesarios en lenguajes estándar de modelado tales como UML , logrando la separación de los componentes del hardware subyacente. En el contexto de la ingeniería de software, el desarrollo dirigido por modelos (MDE) y el enfoque de modelado de dominio específico (DSM) han dado lugar a un cambio de paradigma de "code-centric" a "model-centric". Estos enfoques promueven la sistematización y la automatización de la construcción de artefactos de software. Los modelos son considerados construcciones de primera clase en el desarrollo de software, y el conocimiento de los desarrolladores es encapsulado por medio de transformaciones de modelos. La característica esencial del MDE y DSM es que el principal objetivo del desarrollo de software son los modelos. Su principal ventaja es que los modelos se pueden expresar en diferentes niveles de abstracción y por lo tanto están menos ligados a una tecnología específica. Esto es especialmente relevante para los sistemas de software en el dominio de la computación ubicua, que consisten en aplicaciones dinámicas y distribuidas y corren sobre plataformas heterogéneas de hardware, tales como los sistemas robóticos.

Por último, la arquitectura orientada a servicios (SOA) es un conjunto flexible de principios de diseño utilizados durante las fases de desarrollo e integración de sistemas de computación. Un sistema basado en una arquitectura SOA empaqueta su funcionalidad como un conjunto de servicios interoperables que pueden ser utilizados dentro de múltiples sistemas, pertenecientes a dominios de negocios varios. SOA define cómo integrar aplicaciones muy dispares para un entorno basado en la Web y utiliza múltiples plataformas de ejecución. En lugar de definir una API, SOA define la interfaz en términos de protocolos y funcionalidad. SOA requiere el acoplamiento flexible de servicios con los sistemas operativos, y otras tecnologías que subyacen en las aplicaciones. SOA separa las funciones en unidades distintas, o servicios [12], que los desarrolladores hacen accesibles a través de una red a fin de permitir a los usuarios combinarlos y reutilizarlos en la producción de sus aplicaciones. Estos servicios y sus consumidores se comunican entre sí pasando los datos en un formato común, bien definido, o mediante la coordinación de una actividad entre dos o más servicios [13].

Por lo tanto, los paradigmas CBD y SOA proporcionan un punto de partida para lograr un enfoque MDE en robótica, donde las diferencias entre las distintas plataformas de software y sistemas middleware pueden quedar completamente ocultos al usuario debido a la definición de niveles de abstracción intermedios. Por el momento, no hay ninguna propuesta que tome ventaja de la aplicación combinada de CBD, SOA y MDE para el desarrollo de sistemas de software robóticos en general, ni tampoco para el desarrollo de sistemas robóticos educativos en particular.

5. CONTRIBUCION ORIGINAL

La contribución original de este proyecto consiste en el desarrollo de un marco metodológico equipado con herramientas para la construcción de sistemas de software robóticos, en particular nos hemos centrado en la propuesta educativa de Physical Etoys y Butia [23], pero consideramos que la metodología es de carácter lo suficientemente general para ser aplicada a otros sistemas robóticos también. Hoy en día existen sólo propuestas preliminares sobre la aplicación de MDE en la robótica (ver por ejemplo los trabajos descriptos en [18], [19], [20] y [21]). Ninguna de estas propuestas provecha la combinación del paradigma MDE con enfoques orientados a servicios y basados en componentes, tal como proponemos en este proyecto.

6. RECURSOS HUMANOS

El equipo está integrado por cinco investigadores senior, el Prof. Paul Puleston es actualmente Profesor Titular en la UNLP e investigador del Consejo Nacional de Investigación Científica y Técnica, en el Laboratorio de Electrónica Industrial, Control e Instrumentación (LEICI). Su área de experticia son los sistemas de control automático, teoría y aplicaciones. La Dra. Claudia Pons conduce, desde hace varios años, un grupo de investigación dedicado al estudio de metodologías de desarrollo de software basado en modelos y métodos formales. Actualmente dirige un centro de investigación en la UAI, especializado en robótica (ver http://www.caeti.uai.edu.ar). MSc. Gonzalo Zabala es el director del Laboratorio de Robótica de la Universidad Abierta Interamericana (UAI), ver http://tecnodacta.com.ar/gira/. Es el creador de Physical Etoys y participa activamente

en el programa "Conectar Igualdad" (http://www.conectarigualdad.gob.ar), integrando su Consejo Asesor. Gonzalo Zabala y su grupo tienen amplia experiencia en desarrollo de proyectos educativos utilizando robots. La Dra. Gabriela Arévalo es experta en el campo de la ingeniería de software y específicamente en las metodologías de refactorización/reingeniería. Ella trabaja en el CONICET y contribuye a los proyectos de investigación en la UAI. MSc. Carlos Luna es investigador Principal de la Universidad de la República y de la ANII ("Agencia Nacional de Investigación e Innovación") en Uruguay. Ha participado en numerosos proyectos en el área de los métodos formales aplicados a la ingeniería de software, y en el área de la educación y las tecnologías en la educación.

El resto del equipo está integrado por estudiantes de pregrado y postgrado que trabajan en la robótica y en MDD. Tres tesis de licenciatura en Informática y dos tesis doctorales se están desarrollando en el contexto de este proyecto.

REFERENCIAS

[1] Programa Conectar Igualdad. http://www.conectarigualdad.gob.ar/ (accedido en May 2011)

[2] Physical Etoys. GIRA Grupo de Investigación en Robótica Autónoma del CAETI. http://tecnodacta.com.ar/gira/ (accedido en May 2011)

[3] Bruyninckx, H.: Open robot control software: The OROCOS project. In: Proceedings of 2001 IEEE International Conference on Robotics and Automation (ICRA'01), vol. 3, pp. 2523–2528. Seoul, Korea (2001)

[4] Ando, N., Suehiro, T., Kitagaki, K., Kotoku, T., Yoon, W.K.: RT-middleware: Distributed component middleware for RT (robot technology). In: International Conference on Intelligent Robots and Systems 2005 (IROS 2005), pp. 3933–3938 (2005)

[5] Clemens Szyperski Component Software: Beyond Object-Oriented Programming. 2nd ed. Addison-Wesley Professional, Boston ISBN 0-201-74572-0 (2002).

[6] Microsoft, "Microsoft robotics developer studio," 2009,http://msdn.microsoft.com/en-us/robotics/default.aspx, visited on March 11th 2009.

[7] Barner, S., Geisinger, M., Buckl, C., Knoll, A.: EasyLab: Model-based development of software for mechatronic systems. In: IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications. Beijing, China (2008)

[8] Brooks, A., Kaupp, T., Makarenko, A., Oreback, A., Williams, S.: Towards component-based robotics. In: Proc. of 2005 IEEE/RSJ Int. Conf. on Intellegent Robots and Systems (IROS'05), pp. 163–168. Alberta, Canada (2005)

[9] Nesnas, I., Wright, A., Bajracharya, M., Simmons, R., Estlin, T.: CLARAty and challenges of developing interoperable robotic software. In: Proceedings of 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003), vol. 3, pp. 2428–2435 (2003).

[10] Gerkey, B.P., Vaughan, R.T., Howard, A.: Most valuable player: a robot device server for distributed control. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1226–1231. Wailea, Hawaii (2001) Player Stage

[11] C. Schlegel, ''Communication patterns as key towards component interoperability,'' in Software Engineering for Experimental Robotics (Series STAR, vol. 30), D. Brugali, Ed. Berlin, Heidelberg: Springer-Verlag, , pp. 183–210. Smartsoftware (2007)

[12] Bell, Michael. "Introduction to Service-Oriented Modeling". Service-Oriented Modeling: Service Analysis, Design, and Architecture. Wiley & Sons. pp. 3. ISBN 978-0-470-14111-3. (2008).

[13] Bell, Michael. SOA Modeling Patterns for Service-Oriented Discovery and Analysis. Wiley & Sons. pp. 390. ISBN 978-0470481974. (2010).

[14] Steven Kelly, Juha-Pekka Tolvanen. Domain-Specific Modeling. John Wiley &Sons, Inc. 2008.

[15] Stahl, M Voelter. Model Driven Software Development. John Wiley. (2006).

[16] Claudia Pons, Roxana Giandini, Gabriela Pérez. "Desarrollo de Software Dirigido por Modelos. Teorías, Metodologías y Herramientas", Editorial: McGraw-Hill Education. (2010).

[17] Jacobson, Ivar; Grady Booch; James Rumbaugh. The Unified Software Development Process. Addison Wesley Longman. ISBN 0-201-57169-2. (1998).

[18] Christian Schlegel, Thomas Haßler, Alex Lotz and Andreas Steck. Robotic Software Systems: From Code-Driven to Model-Driven Designs. In procs. Of ICAR 2009. International Conference on Advanced Robotics. IEEE Press (2009)

[19] Iborra, A., Alonso, D., Cáceres, F. Ortiz, J., Sanchez Palma, P. and Alvarez, B.. Design of Service Robots. Experiences Using Software Engineering. IEEE Robotics & Automation Magazine 1070-9932/09/ IEEE MARCH 2009

[20] Barner, S., Geisinger, M., Buckl, C., Knoll, A.: EasyLab: Model-based development of software for mechatronic systems. In: IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications. Beijing, China (2008)

[21] A. Brooks, T. Kaupp, A. Makarenko, S. Williams, and A. Oreback,''Towards component-based robotics'' in Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems, 2005, vol. 1, pp. 163–168. (2005)

[22] Brooks, R.,"A robust layered control system for a mobile robot". Robotics and Automation, IEEE Journal,1986.

[23] "Butiá Project", http://www.fing.edu.uy/inco/proyectos/butia/, visited on August 2011

[24] Murphy R. R., An Introduction to AI Robotics (Intelligent Robotics and Autonomous Agents), Chapters 3-6, MIT Press, 2000.