Demostración de NDT-Driver: una herramienta de soporte a los mecanismos de transformación de NDT

J. A. García-García, M.J. Escalona

[1] Grupo de Investigación IWT2. Universidad de Sevilla, Sevilla, España.
julian.garcia@iwt2.org; mjescalona@us.es

Abstract. En el contexto del paradigma de ingeniería guiada por modelos (MDE), el uso de metodologías –como es el caso de NDT (Navigational Development Techniques) – ayuda a asegurar la calidad de los resultados durante el desarrollo software. Sin embargo, en el día a día de las empresas que trabajan bajo el paraguas de este tipo de metodologías, muy a menudo acaecen problemas que no deberían darse y que provocan en última instancia, que la aplicación de las fases metodológicas sean consideradas como una mera formalidad sin utilidad aparente. Además, la definición teórica de metodologías hace impracticable su aplicación en contextos empresariales debido al uso de una terminología demasiado abstracta (metamodelos, transformaciones, conceptos, etc.). Por esto, se hace necesario desarrollar herramientas de soporte metodológico que oculten esta terminología teórica para mejorar su aplicabilidad. Este artículo presenta una demostración de NDT-Driver, una herramienta de soporte de la metodología NDT que permite a las empresas aprovechar todo el potencial de MDE de una manera transparente, sin tener que conocer conceptos tan abstractos como el de transformación de modelos. Actualmente, esta herramienta está siendo utilizada de forma satisfactoria en entornos reales de diferentes ámbitos de negocio.

Keywords: Model-Driven Engineering, Herramienta Basadas en Modelos, Herramienta, NDT

1 Introducción, contexto y necesidad

El paradigma de ingeniería guiada por modelos [1] (MDE, Model Driven Engineering) surgió con el fin de hacer frente a la complejidad de las plataformas y la incapacidad de los lenguajes de tercera generación para el alivio de esta complejidad. Todo ello con el propósito de sistematizar lo máximo posible el ciclo de vida de desarrollo software. Para alcanzar este objetivo, MDE se basa, como forma primaria de expresión, en la definición de modelos (de diferente nivel de abstracción) y reglas de derivación entre ellos para generar modelos cada vez más específicos y cercanos a la plataforma final. Además, si se desarrolla herramientas eficientes para aplicar este paradigma, el proceso puede ser automatizable en gran parte.

Sin embargo, MDE no es fácil de aplicar en entornos reales debido a que su terminología no resulta atractiva ni comúnmente entendible para los equipos de desarrollo. Conceptos tales como modelos, metamodelos, reglas de derivación o

transformación, entre otros, no son notaciones comunes en el entorno empresarial y se consideran demasiado abstractos y complejos.

Este artículo presenta cómo la metodología NDT [2] (Navigational Development Techniques) aborda esta problemática con su herramienta de soporte NDT-Driver con la que es posible dar respuesta a las dificultades acaecidas durante la especificación de sistemas de información.

2 Visión general de la metodología NDT

Actualmente, NDT proporciona soporte a todas las fases del ciclo de vida software (Estudio de Viabilidad, Ingeniería de Requisitos, Análisis, Diseño, Implementación, Pruebas, y Mantenimiento). Este soporte pasa por la definición formal de un conjunto de metamodelos para definir cada una de sus fases y la definición de reglas de derivación con las que es posible generar modelos de una fase de manera sistemática a partir de otros modelos de otras fases. Por una parte, los metamodelos se formalizan mediante perfiles UML, con el aporte de OCL [5] para la definición de restricciones, y el soporte de la herramienta Enterprise Architect [4] como herramienta de modelado UML. Por otra parte, las reglas de derivación son descritas en lenguaje QVT [3].

Sin embargo, como hemos adelantado anteriormente, esta definición teórica de NDT se antoja impracticable en contextos empresariales debido al uso de una terminología demasiado abstracta. Se hizo, por tanto, necesario desarrollar herramientas que ocultasen esta terminología teórica para mejorar la aplicabilidad de NDT en entornos reales. Esto se logró con NDT-Suite [7], un paquete de herramientas que proporciona soporte al uso de la metodología NDT.

El abanico de herramientas que componen NDT-Suite es muy amplio como para realizar una demostración de forma completa. Por este motivo, este artículo se centra en una de ellas: NDT-Driver. Esta herramienta se describe en la siguiente sección.

3 NDT-Driver

NDT-Driver implementa en Java un conjunto de procedimientos automáticos para ejecutar cada una de las reglas de transformación QVT que define NDT. En este sentido, NDT-Driver es capaz de generar los modelos de la fase de Análisis desde el modelo de la fase de Requisitos, los modelos de la fase de Diseño desde los modelos de Análisis, y el modelo de pruebas de sistema de la fase de Pruebas desde el modelo de Requisitos. Además, la herramienta permite obtener el modelo Requisitos a partir de los requisitos capturados durante la fase de Estudio de Viabilidad del proyecto.

La Figura 1a muestra de la interfaz principal de NDT-Driver, desde la que es posible seleccionar qué transformación llevar a cabo, así como el modo de transformación (mencionado más adelante) deseado.

Asimismo, para facilitar el trabajo en proyectos que siguen ciclos de vida iterativos e incrementales, NDT-Driver permite generar cada modelo del mismo modo, es decir, siguiendo una estrategia incremental. Además, la herramienta permite seleccionar, para cada transformación, el modelo concreto que se desea generar. Por

ejemplo, la Figura 1b muestra la ventana para configurar la transformación de la fase de Requisitos a la fase de Análisis.

(a) Interfaz de Usuario Principal. **(b)** Ventana de Configuración.

Fig. 1. Herramienta NDT-Driver.

La aplicación automática de las reglas de transformación proporciona, sin duda, una posición aventajada al analista a la hora de especificar los distintos modelos de cada fase del ciclo de vida. Sin embargo, estos modelos generados de forma automática deben ser mejorados para completar y enriquecerlos en aras de obtener una versión definitiva.

Durante este proceso de mejora es posible que se detecten necesidades del cliente que no han sido considerados inicialmente, lo que puede conllevar la modificación del modelo origen para volver a generar el modelo en cuestión. Esto puede conllevar la pérdida de todo el trabajo realizado por el analista durante el proceso de mejora.

Como solución, NDT-Driver permite dos modos de transformación: *Reconstrucción* o *Actualización*. El primer modo consiste en volver a generar un modelo desde cero partiendo del modelo origen y descartando el modelo generado, si éste ha sido previamente generado. El segundo modo consiste en transformar sólo aquellos elementos del modelo original que han sido modificados.

Todas las características hacen de NDT-Driver, una herramienta que permite reducir cuantitativamente el tiempo dedicado a la especificación de cualquier proyecto que siga las directrices de la metodología NDT.

En la web del grupo de investigación IWT2[1] (sección "Aprendiendo NDT") el lector puede encontrar diferentes videos de demostración que muestran cómo se utiliza NDT-Driver. Asimismo, el lector puede encontrar en dicha web, un proyecto de prueba (proyecto del Hotel Ambassador) sobre el que ejecutar la herramienta.

4 Conclusiones y lecciones aprendidas

Este artículo presenta NDT-Driver: una herramienta sencilla e intuitiva para que cualquier empresa que desarrolla sus proyectos en el marco de la metodología NDT

[1] http://www.iwt2.org/

pueda aprovechar todo el potencial del paradigma MDE, sin tener que conocerlo teóricamente. De manera concreta, NDT-Driver automatiza todas las reglas de transformación QVT que define NDT

Como lección aprendida de nuestra experiencia [8; 9] en el uso de MDE, podemos concluir que su uso en contextos reales puede mejorar cuantitativamente los resultados del proyecto. De hecho, el uso de perfiles UML, y herramientas basadas en UML ofrecen una interfaz efectiva y atractiva para que cualquier tipo de usuario pueda trabajar de forma eficaz bajo el paraguas de una metodología enmarcada dentro del paradigma MDE (como es el caso de NDT).

Agradecimientos

Este artículo ha sido financiado por el proyecto MeGUS (TIN2013-46928-C3-3-R) del Ministerio de Ciencia e Innovación y el proyecto NDTQ-Framework (TIC-5789) de la Junta de Andalucía (España). Además, también ha sido financiado por la fundación Universia gracias a su programa de becas para estudiantes de doctorado con discapacidad.

Referencias

1. Schmidt, D. C. *Model-Driven Engineering*. IEEE Computer, Computer Society, vol. 39, no. 2, pp. 25-31, 2006.
2. Escalona MJ, Aragón G. *NDT. A model-driven approach for web requirements. IEEE Transactions on software engineering*, vol: 34, pp. 377-394. 2008.
3. OMG. *Documents Associated with Meta Object Facility (MOF) 2.0 Query/View/Transformation*. Object Management Group. URL: http://www.omg.org/spec/QVT/1.0/. 2008.
4. SparxSystems. *Enterprise Architect*. Website: www.sparxsystems.com.au, último acceso abril 2014.
5. ISO/IEC. *ISO/IEC 19507:2012 Information technology - Object Management Group Object Constraint Language (OCL)*. International Organization for Standardization, formal/2012-05-09, 2012.
6. García-García, J.A., Cutilla, C.R., Escalona, M.J., Alba, M., Torres, J. *NDT-Driver, a Java Tool to Support QVT Transformations for NDT*. In the 20th International Conference on Information Systems Development (ISD), DOI 10.1007/978-1-4614-4951-5_8, pp. 170-176., 2012.
7. García-García, J. A., Escalona, M., Domínguez-Mayo, F., Salido, A. *NDT-Suite: A Methodological Tool Solution in the Model-Driven Engineering Paradigm*. Journal of Software Engineering and Applications, 7, 206-217. doi: 10.4236/jsea.2014.74022. 2014.
8. Escalona, M. J., Garcia-Garcia, J. A., Mas, F., Oliva, M., and Del Valle, C. (2013) Applying model-driven paradigm: CALIPSOneo experience. Proceedings of the Industrial Track of the Conference on Advanced Information Systems Engineering 2013 (CAiSE'13), vol. 1017, pp. 25-32. Valencia, Spain.
9. Cutilla, C. R., García-García, J. A., Alba, M., Escalona, M.J., Ponce, J., and Rodríguez, L. (2011) Aplicación del paradigma MDE para la generación de pruebas funcionales; Experiencia dentro del proyecto AQUA-WS. Presented at the 6ª Conferência Ibérica de Sistemas e Tecnologias de Informação. ISBN: 978-989-96247-4-0.