Marco para la Captura de Requisitos de Usabilidad en Entornos de MDD[1]

Yeshica Isela Ormeño, Jose Ignacio Panach, Oscar Pastor

Centro de Investigación en Métodos de Producción de Software ProS.
Universitat Politècnica de València
Camino de Vera s/n, 46022 Valencia, Spain
{yormeno, jpanach, opastor}@pros.upv.es

Abstract. Las aplicaciones requieren cumplir con buenos niveles de usabilidad para que permitan su adecuado y completo uso. Dentro de la comunidad interacción persona-ordenador, existen atributos de usabilidad para medir el grado de usabilidad de los sistemas. Sin embargo, en general, estos atributos no se consideran en la comunidad de la ingeniería del software desde la fase de captura de requisitos. Esto desemboca en aplicaciones poco usables en las que el analista debe invertir mucho esfuerzo para adaptarlas a las necesidades del usuario. Si nos centramos en el paradigma de desarrollo software dirigido por modelos (MDD), la ausencia de propuestas para tratar con atributos de usabilidad desde la fase de captura de requisitos es aún más evidente. Existen modelos de requisitos bien conocidos para capturar funcionalidad y estructura, pero no existe un método para guiar la captura de requisitos de usabilidad. El presente trabajo tiene como propósito crear un marco para capturar e incorporar en el sistema los requisitos de usabilidad dentro de un entorno MDD. El método está compuesto por guías para capturar los requisitos de usabilidad y cómo se incorporan estos requisitos en el proceso de desarrollo MDD.

Keywords: Captura de requisitos, usabilidad, interfaz de usuario, desarrollo dirigido por modelos.

1 Introducción

La usabilidad es considerada como un factor importante en la aceptación de los sistemas por los usuarios. Por tanto es necesario precisar enfoques sistemáticos de modelado y análisis desde las etapas iniciales del software. La ISO 9126-1 [6] identifica la usabilidad como una de las características de la calidad del software. Esta ISO propone medir la usabilidad en base a los atributos de usabilidad, que son elementos medibles en el sistema software. Dichos atributos se agrupan en conjuntos llamados subcaracterísticas.

[1] Este trabajo se ha desarrollado con el soporte del MICINN bajo el proyecto PROS-Req (TIN2010-19130-C02-02) y cofinanciado con FEDER.

El paradigma MDD usa modelos como los artefactos de proceso de producción de software, y desarrolla pasos que consisten en la automatización de las aplicaciones de transformación sobre esos modelos. En la actualidad existen métodos y modelos para la captura de requisitos funcionales y no funcionales. Sin embargo, no existe ninguna propuesta para maximizar los atributos de usabilidad en el proceso de desarrollo MDD. Estos atributos se suelen incorporar al sistema de manera directa en el código generado, rompiendo claramente con el paradigma MDD, que aboga por modelos conceptuales holísticos, donde el modelo conceptual en sí es el sistema [10]. En estos métodos holísticos, el analista no debe implementar nada de código, ya que éste puede ser generado desde el modelo conceptual mediante un compilador de modelos.

La incorporación manual de los requisitos de usabilidad en el código generado tiene una serie de inconvenientes:

- Los requisitos de usabilidad pueden estar relacionados con la arquitectura del sistema y con otros requisitos funcionales. En ese caso, implican cambios muy costosos [5].
- En MDD, el código del sistema se puede generar cada vez que el analista modifique parte del modelo conceptual. La regeneración del código es una ventaja que se pierde si parte del código debe ser implementado a mano, ya que los cambios manuales se tienen que aplicar tras cada generación.
- Las características del sistema representadas en el modelo conceptual son independientes de plataforma, pero el código suele depender. Por tanto, las soluciones son menos reusables, ya que son dependientes de un lenguaje de implementación y de una plataforma.

Todos estos inconvenientes se solucionarían si los entornos MDD trataran los requisitos de usabilidad desde las fases iniciales del proceso de desarrollo [16] (modelo conceptual). Este trabajo presenta una idea emergente para incorporar requisitos de usabilidad dentro de entornos MDD. La propuesta se estructura con las siguientes secciones: en la sección 2 se muestra el estado de arte de la captura de requisitos en el entorno MDD, artefactos, modelos y notaciones revisadas. La sección 3 presenta el método con que se pretende capturar los requisitos de usabilidad en entornos de MDD y los pasos para llevar a cabo la investigación. En la sección 4 se presentan las conclusiones de este trabajo.

2 Estado del Arte

Hasta el momento no se han encontrado trabajos que traten la captura de requisitos de usabilidad desde la definición del modelo conceptual, pero sí varios trabajos que analizan diversas situaciones respecto a tratamientos de requisitos funcionales, no funcionales, propuestas de desarrollo y actividades de captura de requisitos de interacción, todos ellos en torno al enfoque MDD. Por ejemplo, el trabajo de Ameller [2], muestra el estado de arte del enfoque MDD con respecto a los requisitos no funcionales (NFRs). Este autor indica que los NFRs no están dirigidos a métodos MDD, esbozando un marco de ajuste personalizado para integrar los NFRs en el proceso MDD. Otro autor es Fieber [4], quien presenta proyectos de investigación

sobre la evaluación de la usabilidad de modelos y el desarrollo dirigido por modelos en una organización industrial, que después de haber realizado una investigación empírica y cuantitativa, deduce que el modelado y las actividades MDD son ejecutadas, pero rara vez manejadas en un modelo de proceso formal. España [3], plantea propuestas para afrontar la captura de requisitos y el desarrollo de interfaces usuario avanzadas que garanticen la usabilidad con la aplicación de OO-Method en las etapas del proceso de MDD. Otro trabajo fue desarrollado por Panach [12], quien define una actividad de captura de requisitos de interacción para OO-Method haciendo uso de la notación formal CTT [14]. Otro aspecto a considerar es que estos trabajos tratan la usabilidad del sistema durante el proceso de desarrollo software pero no inciden en la captura de requisitos de usabilidad, lo que puede provocar en los usuarios la no satisfacción total en la interacción con el sistema.

Hay otros trabajos centrados en la medición de usabilidad basados en modelos conceptuales. Como por ejemplo Molina [11], quien intenta reducir algunos de los fallos detectados en la usabilidad de las aplicaciones web. Para ello, propone medir los atributos de usabilidad desde la fase inicial del proceso de desarrollo. Estos trabajos tienen como inconveniente que no pueden medir aquellos atributos de usabilidad que son totalmente subjetivos, como el grado de atracción que el sistema provoca en el usuario.

Por otro lado, existen varios trabajos que se han centrado en definir notaciones con sus respectivas herramientas para la captura de requisitos de interacción. Por ejemplo, Akoumianakis [1] muestra una notación gráfica para la gestión de requisitos NfRn (notación de requisitos no funcionales) contando con una herramienta de apoyo llamada GECg. Esta herramienta se usa para especificar atributos de usabilidad como la adaptabilidad, escalabilidad y portabilidad. Otro autor es Kristensen [9], quien define otra notación gráfica y la herramienta NoTICe que ilustra el desarrollo de los casos de interacción que son utilizados en la fase del análisis. Por su parte Smith [17] describe una notación para la captura de requisitos y el uso de una herramienta que lo soporta de manera gráfica, obteniendo así requisitos formales para su conversión en autómatas que son revisados y verificados en tiempo de ejecución. El trabajo de Robles [15], define WebSpec, un artefacto para la captura de requisitos de interacción y características de navegación para aplicaciones web independientes del proceso de desarrollo, mejorando el ciclo de desarrollo en periodos cortos de tiempo.

Las notaciones arriba mostradas tienen en común poseer su propia herramienta y son utilizados solo para algunos atributos de usabilidad, pero no para todos los atributos identificados en la ISO 9126-1 [6]. Solo se enfocan sobre una actividad en particular o cierto tipo de NFR, los artefactos no son utilizados junto con otras técnicas estudiadas.

Revisados estos trabajos centrados en la captura de requisitos de usabilidad podemos indicar, que no existe un método que guíe al analista en la captura de requisitos de usabilidad dentro de un entorno MDD. Los trabajos actuales sólo capturan requisitos de interacción para aplicaciones específicas, sin indicar si esos requisitos mejoran o empeoran la usabilidad del sistema.

3 Propuesta del método para la captura de requisitos de usabilidad en MDD

Antes de abordar la propuesta del método, listamos los pasos a seguir para llevar a cabo nuestra investigación:

a. Realizar una revisión sistemática [8] de aquellos mecanismos de recopilación de información como guías de usabilidad [7], heurísticos y notaciones definidos en la literatura.
b. Definir las plantillas para extraer los requisitos de usabilidad.
c. Mapear los requisitos capturados en uno o varios de los atributos de usabilidad determinados por la norma ISO 9126-1 [6], sean éstos de aprendizaje, comprensión, operatividad, nivel de atracción o conformidad.
d. Análisis y elaboración de método propuesto.
e. Evaluación y comparación de los resultados obtenidos con la aplicación del método.

Con la elaboración e implantación del método para la captura de los requisitos de usabilidad en entornos MDD, se persigue insertar atributos de usabilidad en etapas tempranas del desarrollo software, a un alto nivel de abstracción. Nuestro planteamiento de la propuesta del método está basado en las siguientes fases:

1. Utilizar las plantillas para extraer los requisitos de usabilidad durante las entrevistas con el usuario.
2. Representar la información de las plantillas en un modelo de requisitos mediante una notación establecida.
3. Definir las transformaciones entre el modelo de requisitos obtenido y el método MDD sobre el cual se quiera aplicar la propuesta.
4. Generación de código a partir de los modelos del método MDD mediante un compilador de modelos.

En la Fig. (1) se esquematiza este proceso. La primera acción del método será la utilización de plantillas para la captura de requisitos de usabilidad e interacción. Esto se realizará mediante guías de usuario y heurísticos ampliamente conocidos y utilizados en la captura de requisitos funcionales y no funcionales. Las plantillas tienen como finalidad la captura de requisitos de usabilidad en directo contacto con los stakeholders, (paso1). Posterior a esta etapa y ya estando los atributos de usabilidad mapeados, se inicia la construcción del modelo de requisitos, que consiste en que cada uno de los atributos de usabilidad, extraídos mediante las plantillas y aptos para su posible medición, tendrán una representación en el modelo de requisitos. En este paso se definirá el proceso de cómo representarlos. El modelo de representación elegido podría ser uno de los modelos existentes en la literatura, (paso 2).

El método MDD sobre el que se trabaja tendrá sus propios modelos y notaciones que no se tendrían porque corresponder con el modelo utilizado para representar los requisitos de usabilidad. Lo que implica que en el siguiente paso, se definan las transformaciones entre el modelo de requisitos y los métodos existentes en el entorno

MDD, (paso 3). Como paso final una vez construido el modelo conceptual, se puede generar el código del sistema mediante el compilador de modelos del método MDD (paso 4).

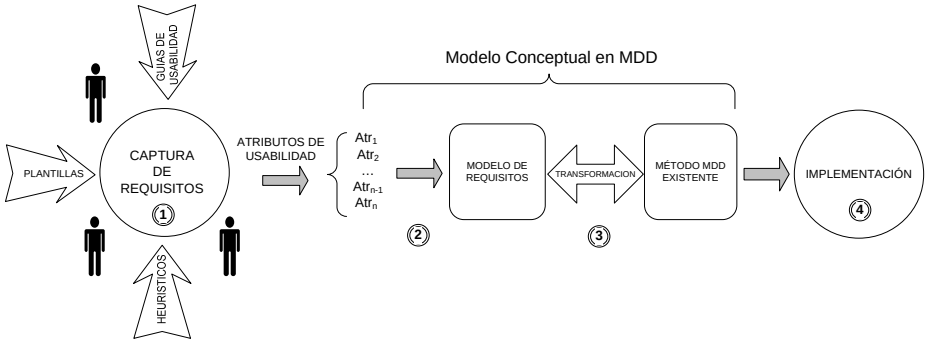

Fig. 1. Proceso de captura de requisitos de usabilidad en MDD.

Como caso práctico, pensamos aplicar esta propuesta a OLIVANOVA [13], una herramienta basada en el concepto de MDD que genera código software de forma automática a partir de un esquema conceptual. Con la propuesta planteada se busca mejorar el proceso de desarrollo software en la etapa crítica de la captura de requisitos, insertando en ella un alto nivel de usabilidad. La generación de plantillas y notación para los atributos de la ISO 9126, contribuyen sustancialmente en la elaboración del modelo conceptual final.

4 Conclusiones

El proceso de captura de requisitos conlleva realizar un análisis crítico tanto de la funcionalidad del sistema como de su interacción con el usuario, incluyendo la usabilidad. Sin embargo, son pocos los trabajos que tratan la usabilidad como un requisito de primer nivel, al igual que los requisitos funcionales. La incorporación de la requisitos de usabilidad desde etapas tempranas del proceso de desarrollo software evitaría futuros cambios en la arquitectura del software para adaptar el sistema a las exigencias del usuario y conseguiría una gran aceptación del producto por parte del usuario. Con la propuesta mostrada sería factible obtener los siguientes avances en cuanto a:

- El apoyo en la implantación de las bases para la construcción de los procesos de software holístico.
- La modificación o ingreso de nuevos requisitos se realizan a un alto nivel de abstracción del modelo conceptual, lo que evitaría cambios en el código, reduciendo costos y tiempo de implementación del software.
- La generación de soluciones reusables, dado que éstas son independientes de plataforma y lenguaje de implementación y están en correspondencia al enfoque MDD.
- La evaluación de los atributos de usabilidad determinando su posterior medición.

Este artículo presenta la idea del trabajo a desarrollar durante los próximos años. Como trabajo futuro, se irá desarrollando el marco para la captura de requisitos de usabilidad paso a paso. El objetivo es elaborar todo este trabajo descrito dentro del contexto de una tesis doctoral.

Referencias

1. Akoumianakis, D., Katsis, A., Vidakis, N.: Non Functional User Interface Requirements Notation (NfRn) for Modeling the Global Execution Context of Tasks. In: Karin, C., Kris, L. (eds.) TAMODIA 2006. LNCS, vol. 4385, pp. 259--274. Springer, Heidelberg (2007)
2. Ameller, D., Franch X., Cabot J.: Dealing with Non Functional Requirements in Model Driven Development. In: 18th IEEE International Requirements Engineering Conference, RE2010, pp. 189--198 (2010)
3. España, S., Panach, J.I., Aquino, N., Valverde, F., Pastor, O.: Propuesta para la Captura de Requisitos y el Modelado de la Interacción en el Marco de MDA. Novática: Revista de la Asociación de Técnicos de Informática, ISSN 0211-2124, Nº. 202 (2009)
4. Fieber, F., Regnat, N., Rumpe, B.: Assessing usability of Model Driven Development in Industrial Project. In 4th European Workshop “From code centric to model centric software engineering: Practices, Implications and ROI (C2M)”, pp. 1--9. University of Twente, Enschede: Centre for Telematics and Information Technology (2009)
5. Folmer, E., Bosch, J.: Architecting for usability: A Survey. Journal of Systems and Software, vol. 70 (1), pp. 61--78 (2004)
6. ISO/IEC 9126-1: Software engineering - Product quality - 1: Quality model (2001)
7. Juristo, N., Moreno, A.M., Sánchez, M.I.: Guidelines for Eliciting Usability Functionalities. IEEE Transaction for Software Engineering, vol. 33(11), pp. 744--758 (2007)
8. Kitchenham, B.Procedures for Performing Systematic Reviews. Keele University Technical Report TR/SE-0401 ISSN: 1353-7776 Australia (2004)
9. Kristensen, B. B.: Notation and Tool for Interaction Cases: Understanding, Designing and Implementing Software-Intensive Systems. In: 13th IASTED International Conference on Software Engineering and Applications, SEA 2009, pp. 53--59 (2009)
10. Mellor, S.J., Clark, A.N., Futagami, T.: Guest Editors' Introduction: Model-Driven Development. IEEE Software, vol. 20, pp. 14--18 (2003)
11. Molina, F., Toval A.: Integrating usability requirements that can be evaluated in design time into Model Driven Engineering of Web Information Systems, Advances in Engineering Software of Web Information Systems 40, pp. 1306--1317 (2009)
12. Panach, J.I., Pederiva, I., España, S., Pastor, O.: Generación Automática de Interfaces a Partir de Patrones Estructurales de Tareas. En Actas de Interacción, Puertollano, España (2006)
13. Pastor, O., Molina, J.: Model Driven Architecture in Practice. Springer, Valencia (2007)
14. Paternò, F.: ConcurTaskTrees: An Engineered Notation for Task Models. In: Diaper, D., Stanton, N., Stanton, N.A., (eds.): The Handbook of Task Analysis for Human-Computer Interaction. Lawrence Erlbaum Associates, London, United Kingdom, pp. 483--501 (2004)
15. Robles, E., Garrigós, I., Grigera, J., Wincler, M.: Capture and Evolution of Web Requirements Using WebSpec, In: Gaedke, M., Grissnikalus, M., Diaz, O. (eds.) ICWE 2009. LNCS, vol. 5648, pp. 136--150. Springer, Heidelberg (2009)
16. Sendall, S., Kozaczynski, W.: Model Transformation: The Heart and Soul of Model-Driven Software Development. IEEE Software 20, pp. 42--45 (2003)
17. Smith, M.H., Havelund, K.: Requirements capture with RCAT. Proceedings of the 16th IEEE International Requirements Engineering Conference, RE'08, Art. Nº. 4685668, pp. 183--192 (2008)