Evaluación empírica sobre eficiencia y satisfacción del impacto de la Ingenieria Dirigida por Modelo

Empiric evaluation about efficiency and satisfaction the impact of Model Driven Engineering

Yulkeidi Martínez Espinosa

Cristina Cachero Castro

Santiago Meliá Beigbeder

Yunieski Martínez Espinosa

Universidad de Ciego de Ávila "Máximo Gómez Báez". Ciego de Avila. Cuba

Universidad de Alicante. Alicante. España

Universidad de Alicante. Alicante. España

Universidad de Ciego de Ávila "Máximo Gómez Báez". Ciego de Avila. Cuba

Resumen

La ingeniería dirigida por modelos afirma tener un impacto positivo en la eficiencia y satisfacción del software. Sin embargo, se han realizado pocos esfuerzos para recopilar evidencias que evalúen sus verdaderos beneficios y limitaciones. El objetivo de esta investigación es comparar la eficiencia y la satisfacción de los desarrolladores Web *junior* durante el desarrollo de la capa de negocio de una aplicación Web 2.0 cuando se utiliza un enfoque de ingeniería centrado en el código (.NET), basado en modelo (UML), o dirigido por modelos (OOH4RIA). Para llevar a cabo la investigación se realiza un diseño factorial completo, experimento intra-sujeto en el que 26 sujetos, divididos en cinco grupos, se les pidió que desarro-

llaran los mismos tres módulos de una aplicación web, cada uno usando un método diferente. Se mide la eficiencia y satisfacción con cada enfoque. Los principales resultados obtenidos es que el uso de las prácticas de ingeniería basadas en el modelo parece aumentar significativamente la eficiencia y la satisfacción de los desarrolladores Web *junior*, independientemente de la aplicación en particular. Sin embargo, las actividades de modelado que no van acompañadas de un entorno de generación fuerte de código fuente hacen que la eficiencia y la satisfacción disminuyan por debajo de las prácticas centradas en el código. Se necesita más experimentación para generalizar los resultados a diferentes poblaciones, lenguajes y herramientas, diferentes dominios y tamaños.

Palabras clave: MDE, experimentación, eficiencia, satisfacción, calidad

Abstract

Model-Driven Engineering claims a positive impact on software efficiency and satisfaction. However, few efforts have been made to collect evidences that assess its true benefits and limitations. The objective of this paper is to compare the efficiency and satisfaction of junior Web developers during the development of the business layer of a Web 2,0 Application when using either a code-centric (.NET), a model-based (UML) or a Model-Driven Engineering approach (OOH4RIA). To develop this investigation was designed a full factorial, intra-subject experiment in which 26 subjects, divided into five groups, were asked to develop the same three modules of a Web application, each one using a different method. We measured their efficiency and satisfaction with each approach. The principals results obtained are: the use of Model-Driven Engineering practices seems to significantly increase both efficiency and satisfaction of junior Web developers, regardless of the particular application. However, modeling activities that are not accompanied by a strong generation environment make efficiency and satisfaction decrease below code-centric practices. Further experimentation is needed to be able to generalize the results to a different population, different languages and tools, different domains and different application sizes.

Keywords: *MDE, experiment, efficiency, satisfaction, quality*

Introducción

La comunidad de Ingeniería de la Web ha dedicado muchos esfuerzos a la definición de enfoques MDE (Valverde & Pastor, 2009), fuertemente caracterizado por la adopción de modelos conceptuales, con el fin de mejorar los procesos de desarrollo de software de aplicaciones Web. Estos enfoques ofrecen altos niveles de abstracción, capturando las características más sobresalientes de las aplicaciones Web, que pueden ser utilizados para la definición de modelos de aplicación abstrayendo los detalles de implementación. MDE también hace énfasis en la importancia de las transformaciones de modelos, lo que lleva a la generación automática del código de la aplicación a partir de los modelos conceptuales de alto nivel. Entre los argumentos más comúnmente aducidos en favor de los enfoques MDE se encuentran: la facilidad de mantenimiento, incremento de la eficiencia y satisfacción de los desarrolladores.

Para consolidar el avance del desarrollo de software MDE, es fundamental proporcionar evidencias empíricas que corroboren o refuten las promesas de mejora que han acompañado a este paradigma desde su incepción y aún más importante ponerlo a disposición de la comunidad de Ingeniería del Software (IS) (Martínez, Cachero, & Meliá, 2012).

La evaluación de los productos de software es una de las actividades que se incluyen dentro del desarrollo y ejecución de cualquier ciclo de vida. Esta evaluación puede realizarse a través de un conjunto de métricas bien definidas que aseguran la calidad del producto de software. De acuerdo al estándar ISO/IEC 25000 (ISO/IEC 25000, 2014), la calidad se define como el "grado en que el producto software satisface las necesidades expresadas o implícitas, cuando es usado bajo condiciones determinadas" y puede evaluarse a partir de dos niveles: 1) la calidad del producto sistema/software y 2) calidad en uso. Específicamente dentro de la calidad en uso se encuentran las sub-características eficiencia y satisfacción.

La eficiencia se define como "recursos gastados (tiempo, esfuerzo, materiales, costo, etcétera) en relación con la precisión e integridad con la que los usuarios alcanzan los objetivos" (ISO/IEC 25010, 2011). Las ganancias de eficiencia son, a menudo, una las principales motivaciones para seleccionar nuevas tecnologías. Mientras la satisfacción, puede ser definida como la "grado del producto software de satisfacer a los usuarios en un contexto especificado de uso" (ISO/IEC 25010, 2011).

Los enfoques de MDE son a menudo reconocidos como una solución para: aumentar la eficiencia y satisfacción de los desarrolladores, así como mitigar la complejidad de tareas de mantenibilidad de software. Sin embargo, no hay pruebas empíricas de sus beneficios y limitaciones con respecto a las prácticas de eficiencia y satisfacción basadas en modelos o centradas en el código. Muchos autores han escrito acerca de la importancia de proporcionar evidencia empírica en la IS (Dyba, Kitchenham, & Jorgensen, 2005); (Kitchenham, et al., 2007); (Martínez, Cachero, & Meliá, 2012); (Mariño & Alfonzo, 2017); que pueden ser proporcionada en forma de encuestas, experimentos, estudios de casos, análisis post-mortem de estudios o revisiones sistemáticas de la literatura; que miden directamente el efecto del paradigma elegido en el rendimiento y la satisfacción del desarrollador (Wohlin, et al., 2012).

Un experimento se define como "las investigaciones donde las posibles variables perturbadoras han sido aleatorizadas" (Kish, 1959); (Mendes, Mosley, & Counsell, 2006), o sea, es un procedimiento mediante el cual se trata de comprobar (confirmar o verificar) una o varias hipótesis relacionadas con un determinado fenómeno, mediante la manipulación de las variables que presumiblemente son su causa. La experimentación (o evaluación experimental) constituye uno de los elementos claves del método científico y es fundamental para poder ofrecer explicaciones causales.

La evaluación empírica de las actividades de eficiencia (anteriormente productividad) y satisfacción en metodologías MDE ha ido ganando impulso en los últimos años. Esta afirmación se apoya en el mapeo sistemático (Martínez, Cachero, & Meliá, 2011) que recoge la evidencia empírica sobre la eficiencia del proceso de desarrollo y mejora de calidad del software debido a la utilización de enfoques MDE durante el período 2001 a 2010.

Teniendo como punto de partida esta situación, el objetivo de este estudio empírico, siguiendo la plantilla de *Goal/Question/Metric* (GQM) (Perry, Porter, & Votta, 2000), es comparar la eficiencia y la satisfacción de los desarrolladores Web *junior* durante el desarrollo de la capa de negocio de una aplicación Web

2.0 cuando se utiliza un enfoque de ingeniería centrado en el código (.NET), basado en modelo (UML), o dirigido por modelos (OOH4RIA). El contexto del estudio es un grupo de estudiantes de maestría que desarrollan la capa de negocio de una aplicación Web 2.0 en la Universidad de Alicante. Las preguntas de investigación que permiten respaldar el objetivo de esta experimentación son:

- RQ1: ¿Es la eficiencia del equipo significativamente diferente entre los métodos, sin considerar el módulo de la aplicación que están desarrollando?
- RQ2: ¿Es la satisfacción de los desarrolladores significativamente diferente entre los métodos, sin considerar el módulo de la aplicación que están desarrollando?

Materiales y métodos o Metodología computacional

Para responder a las preguntas de investigación, se definen las siguientes variables o factores independientes (VI), manipuladas experimentalmente: Meth (método, variable categórica con tres niveles: *code-centric*, MBD, MDE) y Mod (módulo, una variable categórica con tres valores posibles: grupos, eventos y organización de una aplicación Web 2.0). Las variables dependientes (VD) son: la eficiencia del equipo con cada método y módulo -E(Meth, Mod)- y la satisfacción de los desarrolladores -S(Meth, Mod)-. Las medidas han sido usadas para probar las siguientes hipótesis:

- HE (Eficiencia): E(MDE)>E(MBD)>E(*code-centric*), es decir, los equipos desarrolladores son significativamente más productivos con el método MDE (OOH4RIA), seguido del método MBD (UML), seguido del *code-centric* (.NET).
- HS (Satisfacción): S(MDE)>S(MBD)>S(*code-centric*), es decir, los desarrolladores están significativamente más satisfechos con el método MDE (OOH4RIA), seguido del método MBD (UML), seguido del *code-centric* (.NET).

El experimento se realiza con 30 individuos (seleccionado por conveniencia) que cursaban la maestría Desarrollo de Aplicaciones Web de la Universidad de Alicante, divididos en seis grupos de 4 o 6 personas, que desarrollan tres módulos, cada uno aplicando diferentes métodos. Al final de las observaciones solo quedaron 5 grupos pues un grupo de 4 no terminó el experimento por razones de trabajo. La muestra final es de 26 (25 hombres y 1 mujer), con más de 2 años de experiencia profesional como desarrolladores de aplicaciones Web, edad media de 25,6 años y todos ellos graduados de Ingeniería Informática de la Universidad de Alicante. En la tabla 1 se refleja el diseño experimental.

Tabla 1. Diseño experimental: diseño factorial e intra-sujetos.

Equipo-Módulo	Aplicación	Grupos	Eventos	Organización
1	Viajes	*code-centric*	MBD	MDE
2	Eventos	*code-centric*	MDE	MBD
3	Hospitales	MBD	MBD	*code-centric*
4	Académico	MDE	MDE	*code-centric*
5	Facework	MBD	*code-centric*	MDE
6*	Automóvil	MDE	*code-centric*	MBD

El grupo con * no terminó el experimento

El diseño del experimento se basa en el conocido *framework* de experimentación sugerido por Wohlin (Wohlin, Runeson, & Höst, 2000), mientras que los análisis estadísticos se realizan con el PASW *Predictive Statistics Application* v18 (Norusis & et al.).

Resultados y discusión

Para el análisis estadístico, se aplica el 3∗5 *Mixed Design* ANOVA -dado el alto grado de robustez para escalas nominales y ordinales (Norman, 2010)- para un diseño factorial, en los cuales el módulo (grupos, eventos y organización) se considera como variables entre-sujetos, y la E y S calculada por cada método se considera la variable intra-sujeto (indistintamente). Los resultados de la estadística descriptiva -medias y desviación estándar (DS)- aparecen en Tabla 2.

Tabla 2. Estadística descriptiva de las variables S y E

Variables	***code-centric***		**MDB**		**MDE**	
	Media	**DS**	**Media**	**DS**	**Media**	**DS**
HE	0,80	0,29	2,30	1,10	4,60	1,17
HS	4,17	0,72	3,48	0,96	4,76	0,73

Los resultados del análisis estadístico indican que en ninguna de las hipótesis probadas se detecta que la interacción entre Mod*Meth es significativa, pasando a probar cada variable (Mod y Meth) independientemente, sin peligro de modificar los resultados por la existencia de una interacción significativa.

Para darle respuesta a la primera pregunta de investigación (RQ1: Impacto del método sobre la eficiencia del equipo), y en aras de garantizar que aplicar este método estadístico tiene sentido, primero se verifica que no se viole el principio de esfericidad aplicando la prueba W de Mauchly ($W = 0{,}005$; $p > 0{,}05$), (Mauchly, 1940). Los resultados muestran que MDE produjo una eficiencia más alta, seguida de MBD y luego *code-centric* (ver Tabla 1), y que estas diferencias son significativas ($F = 25{,}395$; $p = 0{,}001$). En cambio, el efecto principal del módulo no fue significativo ($F = 0{,}538$; $p > 0{,}05$). Es decir, las diferencias en P están significativamente afectadas por el método usado, sin considerar el módulo especial desarrollado.

En cuanto a la RQ2: Impacto del método sobre la satisfacción del desarrollador, de nuevo la prueba W de Mauchly ($W = 0{,}838$; $p = 0{,}142$), (Mauchly, 1940), demuestra la no violación del principio de esfericidad. Luego, los resultados revelan diferencias significativas ($F = 18{,}04$; $p < 0{,}01$), donde MDE produjo los resultados más altos, seguidos del desarrollo *code-centric* y por último MBD. Por otro lado, se puede observar que la influencia de Mod no fue significativa ($F = 0{,}167$; $p > 0{,}05$). Las diferencias en S están significativamente afectadas por el método usado, sin considerar el módulo especial desarrollado.

Una cuestión fundamental concerniente a los resultados de un experimento es cuán válidos son estos resultados. En general una validez adecuada se refiere a que los resultados son válidos para la población de interés, como mínimo los resultados deben ser válidos para a la población de la que se extrajo la muestra. Cook, Campbell & Day (1979) se delimitan cuatro tipos de amenazas de validez de los estudios empíricos: interna, externa, de constructo y de conclusión. En el presente estudio empírico se ha tenido en

cuenta estas amenazas, que permiten valorar las condiciones en las que son aplicables los experimentos, los beneficios que ofrece y bajo qué circunstancias podrían fallar.

- La amenaza a la validez de conclusión se refiere a la relación entre los tratamientos y las salidas. Para minimizar estas amenazas, se ha intentado capturar la mayor cantidad de medidas como fue posible automáticamente, con la ayuda de sistemas de rastreos (seguimiento) conocidos como JIRA o SVN. Adicionalmente los test estadísticos han sido seleccionados conservadoramente, sin hacer ningún tipo de suposición sobre la distribución de las variables. Sin embargo, el hecho de que los estudiantes reportaron sus propias medidas, junto con la duración del experimento (seis semanas) y el bajo número de sujetos, limitan la validez de conclusiones.
- La amenaza a la validez interna, concerniente con la posibilidad de que existan factores escondidos que puedan comprometer la conclusión, que es efectivamente el tratamiento que causa las divergencias en el resultado. Todos los grupos aplicaron todos los tratamientos a módulos diferentes en distintos momentos, lo que minimiza muchas amenazas internas como la selección, la historia, la maduración o amenazas sociales como la rivalidad compensatoria o la desmoralización resentida. Sin embargo, siendo un diseño intra-sujetos, otros efectos podrían haber ocurrido.
- La amenaza a la validez del constructo se refiere a la relación entre la teoría y la observación. En este sentido los tratamientos y las medidas para valorar la eficiencia y la satisfacción han sido ampliamente usadas en la literatura. No obstante, queda la posibilidad de una interacción de la prueba y los tratamientos, es decir, la necesidad de que los propios estudiantes informaran sobre ciertas medidas podrían haber cambiado su comportamiento. Por otro lado, el hecho de que el experimento tomó más de seis semanas minimiza este riesgo, ya que es muy difícil mantener un comportamiento "potencialmente anormal" durante un largo período de tiempo sin ser detectado. También, las hipótesis del experimento (es decir, mayor eficiencia y satisfacción en ambientes MDE) eran muy fáciles de adivinar, así que los estudiantes podrían haberse sentido influenciados a reportar menos tiempo cuando usan MBD o MDE. De todos modos, los observadores del experimento tuvieron especial cuidado para no descubrir estas hipótesis a los estudiantes. Adicionalmente, el experimento se ve afectado por una generalizabilidad restringida entre los conceptos: se ha verificado un resultado positivo entre la eficiencia y MDE, pero no se puede garantizar que no se vea afectado por otras características del software desarrollado, como la modularidad, la reuzabilidad, o cualquier otro atributo de calidad.
- Por último, la amenaza de validez externa está relacionada con la generalizabilidad de los resultados. En este grupo se han identificado un conjunto de amenazas como: falta de representatividad de la muestra (estudiantes de maestría), ambiente académico, una arquitectura estricta y un dominio y complejidad restringida. También, debido al acoplamiento existente entre el método y la herramienta, todos los métodos fueron acompañados por herramientas específicas. Aunque se trató de seleccionar ambientes de desarrollo ampliamente conocidos y -cuando fue posible- usar estándares (por ejemplo, UML para la actividad de modelado en MBD y MDE), el uso de las diferentes herramientas añade un "sabor especial" a los métodos. Por lo tanto, este experimento tiene que ser reproducido para asegurar que es el método usado y no la herramienta la causa de las diferencias observadas.

Conclusiones

En este documento, se presenta un análisis riguroso de un experimento llevado a cabo en un entorno controlado. La información recopilada muestra que la eficiencia y la satisfacción de los desarrolladores Web *junior* se ven significativamente afectadas por el método de desarrollo, pero son independientes del módulo particular que se está desarrollando. Las principales conclusiones de nuestro estudio (que aún deben ser corroboradas con más repeticiones) son:

1. El enfoque MDE parece aumentar significativamente la eficiencia de los desarrolladores con respecto tanto al MDB como al enfoque centrado en el código (*code-centric*).

2. El enfoque MDE satisface al máximo las expectativas de los desarrolladores *juniors*, seguido por *code-centric* y, en tercer lugar, el enfoque MDB.

Estos resultados están bien alineados con la suposición de que las técnicas basadas en modelos mejoran la eficiencia y también la satisfacción entre los desarrolladores cuando están acompañados por un entorno de generación de código. Sin embargo, la eficiencia y la satisfacción pueden disminuir incluso por debajo de las prácticas centradas en el código cuando las actividades de modelado se utilizan exclusivamente como planos para mejorar la comprensión, y los desarrolladores deben implementar manualmente casi todo el código final.

Además, se necesita más experimentación para separar el efecto de los métodos del efecto de los entornos de desarrollo que los acompañan (Visual Studio 2010, RSM o la herramienta OOH4RIA) y para poder generalizar los resultados a una población diferente, diferentes métodos e lenguajes, diferentes tipos o tamaños de aplicaciones.

Agradecimientos

Este experimento fue realizado en colaboración con el departamento de Lenguajes y Sistemas Informáticos de la Universidad de Alicante. Agradecer, además, a los estudiantes que tomaron su tiempo en participar en el experimento.

Referencias

Cook, T., Campbell, D., & Day, A. (1979). *Quasi-experimentation: Design & analysis issues for field settings.* Houghton Mifflin Boston.

Dyba, T., Kitchenham, B., & Jorgensen, M. (2005). Evidence-based software engineering for practitioners. *Software IEEE, 22*(1), 58–65.

ISO/IEC 25010 (2011). ISO/IEC 25010, Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models. International Standard. First edition 2011-03-01.

ISO/IEC 25000 (2014). ISO/IEC 25000: Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE. International Organization for Standardization, Second edition 2014-03-15. Geneva, Switzerland.

Kish, L. (1959). Some statisticals problems in research design. *American Sociological Review, 24*, 328–338.

Kitchenham, B., Budgen, D., Brereton, P., Turner, M., Charters, S., & Linkman, S. (2007). Large scale software engineering questions-expert opinion or empirical evidence? *Software, IET, 1*(5), 161-171.

Mariño, S.I., Alfonzo, P. L. (2017). Ingeniería de software basado en evidencia: soportes como producto académico. *Enl@ce: revista Venezolana de Información, Tecnología y Conocimiento, 14* (1), 87-96. Universidad del Zulia.

Martínez, Y., Cachero, C., & Meliá, S. (2011). Evidencia empírica sobre mejoras en eficiencia y calidad mediante el uso de aproximaciones MDD: un mapeo sistemático de la literatura. *Revista Española de Innovación, Calidad e Ingeniería del Software., 7*(2).

Martínez, Y., Cachero, C., & Meliá, S. (2012). Mdd vs. traditional software deveopment: A practitioner's subjective perspective. *Information and Software Technology*. doi:http://dx.doi.org/10.1016/j.infsof.2012.07.004

Mauchly, J. W. (1940). Significance test for sphericity of a normal n-variate distribution. *The Annals of Mathematical Statistics, 11*(2), 204–209.

Mendes, E., Mosley, N., & Counsell, S. (2006). The Need for Web Engineering: An Introduction. *Web Engineering*, (págs. 1–27).

Norman, G. (2010). Likert scales, levels of measurement and the laws of statistics, Advances in health sciences education. *15*(5), 625–632.

Norusis, M., & et al. (s.f.). *Pasw statistics 18 guide to data analysis.* Prentice Hall Press.

Perry, D., Porter, A. A., & Votta, L. (2000). Empirical studies of software engineering: a roadmap. *The future of Software Engineering* (págs. 345–355). ACM.

Valverde, F., & Pastor, O. (2009). Facing the Technological Challenges of Web 2.0: A RIA Model-Driven Engineering Approach. *Web Information Systems Engineering-WISE 2009* , (págs. 131–144).

Wohlin, C., Runeson, P., & Höst, M. (2000). *Experimentation in software engineering: an introduction.* Springer Netherlands.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C and Regnell, B. & Wesslën, A. (2012). *Experimentation in software engineering*. Springer Publishing Company.