Theoretical Method Based on MDA to Develop Management Systems of Fairs and Events.

Sandy Suárez Jiménez, Ing.[1], Juan Carlos Mejias Cruz, Ing.[1]
[1]Universidad de las Ciencias Informáticas (UCI), Cuba, ssjimenez@uci.cu, jmejias@uci.cu.

Abstract– ***MDA is an architecture for the development of model-driven software, whose main characteristic is defining formal models and corresponding transformations from one model to another, so that they can be automated. In this work a theoretical method based on MDA to develop management systems of fairs and events, which extends from high-level business models to platform specific models are proposed allowing to reap the benefits that this architecture provides. For its definition, an analysis of the main models of software development methodologies proposed for the specification of the functionality and further transformation at different levels of the architecture is performed. The use of this method will improve the productivity and quality of software development process and enhance the quality of the products developed.***

Keywords-- ***Model-driven architecture, software development, systems management of fairs and events.***

Digital Object Identifier (DOI):
http://dx.doi.org/10.18687/LACCEI2015.1.1.014
ISBN: 13 978-0-9822896-8-6
ISSN: 2414-6668

Método teórico basado en MDA para el desarrollo de sistemas de gestión de ferias y eventos.

Sandy Suárez Jiménez, Ingeniero en Ciencias Informáticas[1], Juan Carlos Mejias Cruz, Ingeniero en Ciencias Informáticas[2].
[1]Universidad de las Ciencias Informáticas (UCI), Cuba, ssjimenez@uci.cu,
[2]Universidad de las Ciencias Informáticas (UCI), Cuba, jmejias@uci.cu.

RESUMEN

MDA es una arquitectura para el desarrollo de software dirigido por modelos, que tiene como característica fundamental la definición de modelos formales y las correspondientes transformaciones de un modelo a otro, de forma que estas puedan ser automatizadas. En el presente trabajo se propone un método teórico basado en MDA para el desarrollo de sistemas de gestión de ferias y eventos, que se extiende desde los modelos de negocio de alto nivel hasta los modelos específicos de plataformas, permitiendo aprovechar los beneficios que dicha arquitectura proporciona. Para su definición se realizó un análisis de los modelos que las principales metodologías de desarrollo de software proponen para la especificación de las funcionalidades y su posterior transformación en los diferentes niveles de la arquitectura. La utilización de este método permitirá mejorar la productividad y calidad del proceso de desarrollo de software, así como elevar la calidad de los productos desarrollados.

Palabras claves: Arquitectura dirigida por modelos, desarrollo de software, sistemas de gestión de ferias y eventos.

I. INTRODUCCIÓN

En la actualidad los sistemas de gestión de ferias y eventos resultan de gran utilidad para las instituciones que los organizan, ya que facilitan la gestión de todo tipo de eventos de una forma más eficiente. Tienen como objetivo fundamental garantizar su satisfactorio desarrollo a partir de la adecuada organización antes de la ejecución y están guiados por dos procesos fundamentales: la convocatoria y la contratación.

El proceso convocatoria tiene como objetivo mantener un registro de los eventos que han sido organizados, así como la información fundamental asociada a los mismos. Es el encargado de la realización de una correcta convocatoria del evento, desde el momento de su creación, a las entidades potencialmente interesadas en participar.

El proceso contratación tiene como objetivo registrar los acuerdos contraídos entre la entidad participante en el evento y la entidad organizadora, así como las obligaciones de pago en caso de existir. Una vez lanzada la convocatoria de un evento determinado, se registran los datos de las entidades interesadas en participar mediante un formulario de inscripción, realizándose posteriormente la negociación del contrato para el
cual se trata de llegar a un acuerdo entre el participante y el organizador del evento.

El desarrollo dirigido por modelos surge como un nuevo paso en el camino hacia una verdadera industrialización de la producción de software. Tras el éxito de la tecnología orientada a objetos, el uso sistemático de modelos se presenta ahora como la forma apropiada para conseguir programar con un nivel más alto de abstracción y de aumentar el nivel de automatización.

Teniendo como premisa que este nuevo paradigma posibilita la especificación del sistema separando la lógica de los detalles de la implementación en una plataforma concreta, surge la necesidad de proponer un método teórico basado en MDA para el desarrollo de sistemas de gestión de ferias y eventos, que permita mejorar la productividad y calidad del proceso de desarrollo de software así como elevar la calidad de los productos desarrollados.

II. MATERIALES Y MÉTODOS O METODOLOGÍA COMPUTACIONAL

Arquitectura Dirigida por Modelos (MDA)

El Object Management Group (OMG) es un consorcio internacional dedicado al cuidado y el establecimiento de diversos estándares para un amplio rango de tecnologías. Uno de los principales objetivos del OMG desde sus inicios en

1989, ha sido ayudar a los usuarios de ordenadores a solventar los problemas de integración e interoperabilidad entre sus sistemas, proporcionando para ello especificaciones abiertas e independientes de fabricantes. Así, persiguiendo este objetivo, a fines del año 2000, la OMG introduce MDA, una arquitectura para el desarrollo de software dirigido por modelos [1].

La característica fundamental de MDA es la definición de modelos formales como elementos de primera clase para el diseño e implementación del sistema y la definición de transformaciones de un modelo a otro, de forma que estas transformaciones puedan ser automatizadas. MDA propone la especificación del sistema, separando la especificación de la lógica, de los detalles de su implementación en una plataforma concreta. Así, MDA y los estándares que soporta, permite que la lógica de los sistemas, expresada en modelos, sea realizada en diversas plataformas específicas a través de la definición de transformaciones.

De este modo, MDA proporciona a las organizaciones una manera de solucionar los problemas de integración de sus sistemas actuales con las nuevas plataformas que surgen para su implementación, a la vez que le permite preservar la lógica de sus aplicaciones y por tanto sus inversiones en plataformas existentes.

En las propuestas descritas en [2,3,4] se demuestra el impacto positivo de la aplicación de aproximaciones MDA en el desarrollo de software. Especialmente en [5] se realiza un estudio de otras propuestas donde se recogen evidencias empíricas del efecto de la aplicación de dicha arquitectura. Por lo anteriormente expresado se adopta MDA como base del método que se propone en este trabajo.

Metodologías analizadas

Una metodología de desarrollo de software es un conjunto de pasos y procedimientos que deben seguirse para desarrollar y documentar un software. Se encarga de elaborar estrategias; centradas en las personas o los equipos, orientadas hacia la funcionalidad y la entrega. Su objetivo es elevar la calidad del software a través de un mayor control sobre el proceso de desarrollo [7].

En la actualidad existen dos grandes tendencias de metodologías: ágiles y tradicionales.

Las metodologías ágiles se caracterizan por tener un desarrollo incremental para producir tempranamente pequeñas entregas en ciclos rápidos, disposición para el cambio y generar pocos artefactos [6]. Entre las más utilizadas se encuentran eXtreme Programming (XP) y Scrum.

Extreme Programming (XP): Creada por Kent Beck, se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, teniendo como objetivo fundamental producir rápidamente versiones del sistema a desarrollar. Además brinda una mayor flexibilidad para reducir riesgos técnicos y manejar cambios en los requisitos con relación a otras metodologías ágiles analizadas [8].

Scrum: Desarrollada por Ken Schwaber, Jeff Sutherland y Mike Beedle. Está especialmente indicada para proyectos con un rápido cambio de requisitos. Como característica principal se encuentra el desarrollo de software mediante iteraciones, denominadas sprints donde el resultado de cada sprint es un incremento ejecutable que se muestra al cliente [9].

Las metodologías tradicionales o pesadas se centran en el control del proceso y establecen una rigurosa planificación de tareas y responsabilidades, generando gran cantidad de documentos y artefactos. Estas se distinguen por especificar las herramientas y notaciones que deben ser utilizadas durante la construcción del producto [6]. Entre las metodologías tradicionales o pesadas más usadas se encuentran RUP (Rational Unified Process) y MSF (Microsoft Solution Framework).

Microsoft Solution Framework (MSF): Es una metodología flexible e interrelacionada con una serie de conceptos, modelos y prácticas de uso, que controlan la planificación, el desarrollo y la gestión de proyectos tecnológicos. Se centra en los modelos de proceso y de equipo dejando en un segundo plano las elecciones tecnológicas. Entre los modelos que propone para planificar las diferentes partes implicadas en el desarrollo de un proyecto se encuentran Modelo de Arquitectura del Proyecto, Modelo de Equipo, Modelo de Proceso, Modelo de Gestión del Riesgo, Modelo de Diseño de Proceso y finalmente el Modelo de Aplicación [11].

Rational Unified Process (RUP): Es una metodología muy utilizada por equipos de desarrollo de software y adaptable prácticamente a cualquier proyecto. Toma en cuenta las mejores prácticas en el modelo de desarrollo de software entre las que se encuentran el desarrollo de software de forma iterativa, el manejo de requerimientos, utiliza la arquitectura basada en componentes, modela el software

visualmente, verifica la calidad del software y controla los cambios [10].

Método teórico basado en MDA para el desarrollo de sistemas de gestión de ferias y eventos

Para la definición del método se tuvo en cuenta las características de cada uno de los niveles definidos por la arquitectura y los modelos propuestos por las diferentes metodologías de desarrollo de software analizadas.

En función del nivel de abstracción, MDA define tres niveles conceptuales de modelado:

En un nivel, se modelan los requisitos del sistema mediante los Modelos Independientes de la Computación (CIM, Computer Independent Model) que sirven de puente entre los expertos del negocio y los desarrolladores que afrontan la realización del sistema. Los modelos de este nivel no representan detalles del sistema en sí, sino más bien del dominio de aplicación del mismo. El método que se presenta en este trabajo se centra en el comportamiento de los sistemas de gestión de ferias y eventos. Por ello, propone la realización de dos modelos a nivel CIM, cuyo objetivo es describir el negocio en el que se desenvuelve el sistema e identificar las funcionalidades que deben ser implementadas. Tales modelos son el Modelo de Procesos del Negocio y el Diagrama de Actividades los cuales se describen brevemente a continuación.

Modelo de Procesos del Negocio: es una parte esencial de cualquier proceso de desarrollo de software. Provee una descripción de dónde se va a ajustar el sistema de software considerado dentro de la estructura organizacional y de las actividades habituales. Captura las actividades manuales y los procedimientos automatizados habituales que se incorporarán al nuevo sistema, con costos y beneficios asociados.

Diagrama de actividades: es un grafo (grafo de actividades) que contiene estados en que puede hallarse una actividad describiendo el orden de las tareas o actividades que logran los objetivos del negocio.

Otro nivel comprende los Modelos Independientes de la Plataforma (PIM, Platform Independent Model), que se utilizan para la representación de las funcionalidades y estructura del sistema, abstrayéndose de los detalles tecnológicos de la plataforma en la que se implementará. Una buena forma de definir los PIM es pensar en el desarrollo del sistema para una máquina virtual de tecnología neutral, que proporciona servicios que se implementarán luego de forma diferente en cada plataforma concreta. Además, los modelos de nivel PIM pueden ser refinados tantas veces como se quiera hasta obtener una descripción del sistema con el nivel de claridad y abstracción deseado. En este nivel el método propone la realización de diversos modelos basados en las funcionalidades y procesos necesarios para un correcto funcionamiento del sistema, ya identificados. Entre estos modelos se encuentran el Modelo de Caso de Uso, el Modelo de Iteración y las Interfaces de Usuario. A continuación se ofrece una breve descripción de cana uno de estos modelos.

Modelo de Caso de Uso: documentan el comportamiento de un sistema desde el punto de vista del usuario. Por lo tanto los casos de uso determinan los requisitos funcionales del sistema, es decir, representan las funciones que un sistema debe ejecutar. Su ventaja principal es la facilidad para interpretarlos, lo que hace que sean especialmente útiles en la comunicación con el cliente.

Modelo de Clases del Análisis: representan los conceptos en un dominio del problema, es decir, se centran en los requisitos funcionales y son evidentes en el dominio del problema porque representan conceptos y relaciones del dominio. Tienen atributos y entre ellas se establecen relaciones de asociación, agregación/composición, generalización/especialización y tipos asociativos. RUP propone clasificar a las clases en Interfaz, de Control y Entidad. Esta clasificación da robustez al modelo porque los cambios tienden a afectar a un área en específico.

Interfaces de Usuario: el diseño de interfaces de usuario es una tarea que ha adquirido relevancia en el desarrollo de software. Descomponiendo un sistema en capas lógicas según su función, se puede ver a la interfaz de usuario como la capa lógica encargada de dar soporte al diálogo con el usuario. Sus funciones, son básicamente dos:

- Entrada: adquirir las órdenes, información, comandos expresados por el usuario a través de diversos dispositivos de interacción.
- Salida: presentar resultados, retroalimentación y cooperar para facilitar al usuario la realización de tareas que pretende resolver el sistema [12].

Para el diseño de las interfaces se propone la utilización del lenguaje de modelado UML pues permite capturar la

variedad de características del sistema en sus correspondientes modelos.

Finalmente, se encuentran los Modelos Específicos de la Plataforma (PSM, Platform Specific Model), que combinan las especificaciones contenidas en un PIM, con los detalles de la plataforma elegida. A partir de los distintos PSM se pueden generar automáticamente distintas implementaciones del mismo sistema. En este nivel el método propone realizar el Modelo de Clases de Diseño, el Modelo de Datos y el Modelo de Implementación.

Modelo de Clases del Diseño: es una abstracción del Modelo de Implementación y su código fuente, y fundamentalmente se emplea para representar y documentar su diseño. Es usado como entrada esencial en las actividades relacionadas a la implementación y representa a los casos de uso en el dominio de la solución.

Modelo de Datos: es usado para describir la representación lógica y física de la información persistente manejada por el sistema.

Modelo de Implementación: El Modelo de Implementación representa la composición física de la implementación en términos de subsistemas de implementación y elementos de implementación. Describe cómo los elementos de diseño se implementan en componentes [13]. Según los autores en [10], se considera el artefacto más significativo del flujo de trabajo de Implementación, debido a la importancia que tiene para los desarrolladores comprender el funcionamiento del sistema desde el punto de vista de componentes y sus relaciones.

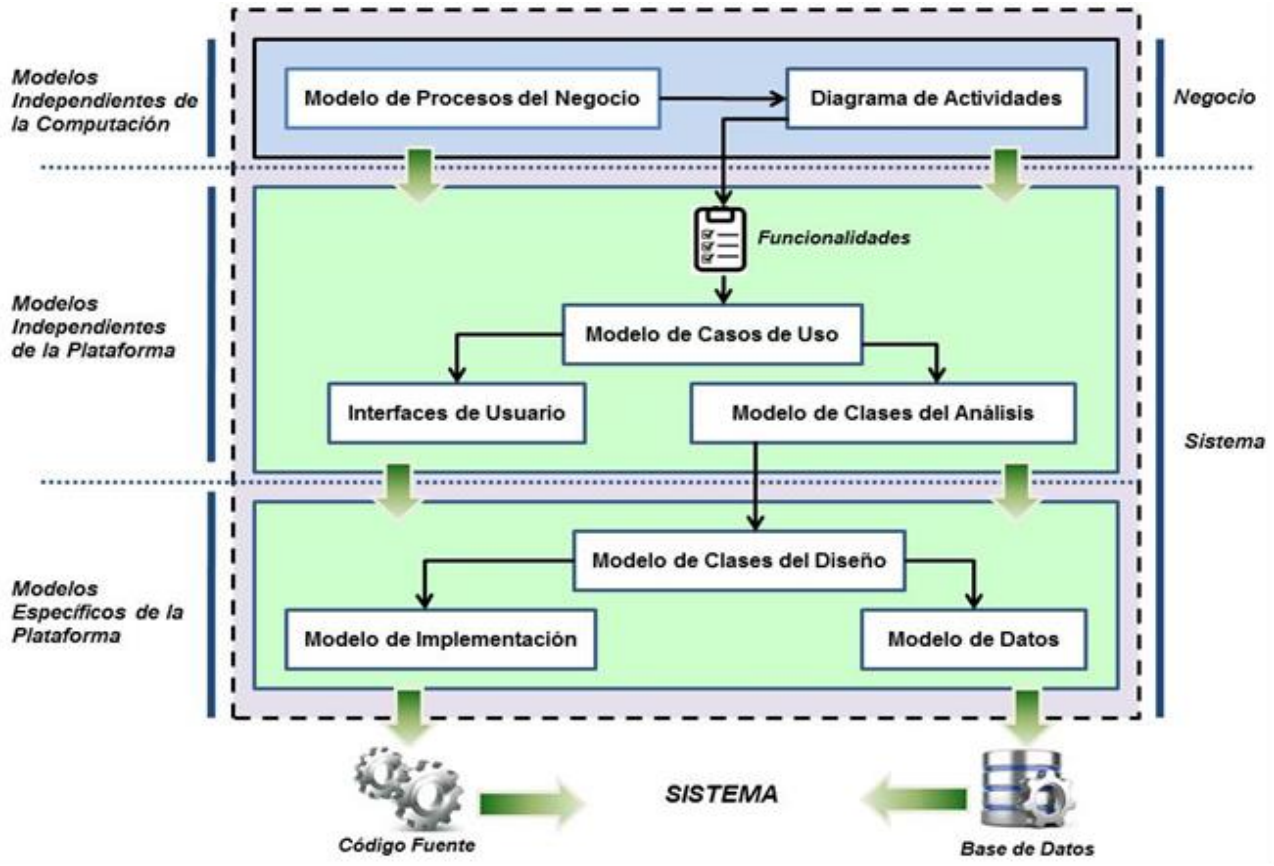
Es imprescindible señalar que los modelos que se describen, no incluyen detalles sobre una tecnología concreta para la implementación de sistemas de gestión de ferias y eventos, pues actualmente existen varias tendencias de desarrollo de dichos sistemas. Sin embargo, tales modelos podrán ser posteriormente refinados, dando lugar a nuevos modelos de nivel PSM dependientes ya de la tecnología seleccionada, como la plataforma .NET (Microsoft .NET), Java, etc. Tanto este último nivel de detalle dentro de los modelos de nivel PSM como la generación de código a partir de ellos quedan fuera del alcance de este trabajo, y se incluyen como líneas de investigación en las que hoy se trabaja.

La Fig. 1 muestra una vista general de la propuesta, donde se muestran los modelos propuestos a cada nivel.

III. Resultados y discusión

El principal resultado de este trabajo es la definición teórica de un método basado en MDA

Figura 1. Vista general de la propuesta basada en MDA. Fuente: Elaboración propia



para el desarrollo de sistemas de gestión de ferias y eventos. Como continuación del mismo, existen diversas líneas de investigación que quedan abiertas y en las que se está trabajando. Tales líneas están relacionadas directamente con el método planteado y son el resultado de los análisis realizados durante la definición del mismo.

En primer lugar, una de las principales líneas de investigación está relacionada con la identificación de los tipos de transformaciones a utilizar entre cada uno de los modelos
propuestos (automatizadas, semi-automatizadas o manuales), pues uno de los principales esfuerzos de la comunidad de investigadores es lograr que las transformaciones se realicen de la forma más automatizada posible, garantizando la calidad de los modelos.

Se debe garantizar una correcta validación de los modelos antes y después de realizar transformaciones, utilizando para ellos algunas de las técnicas definidas.

También se pretende implementar una herramienta que una vez especificada las reglas de transformaciones utilizando los lenguajes definidos para ello, realice dichas transformaciones de forma automatizada.

Una vez desarrolladas estas líneas propuestas, el método resultante permitirá mejorar la productividad y calidad del proceso de desarrollo de software, así como elevar la calidad de los productos desarrollados. Lo expuesto anteriormente se puede evidenciar en [14], [15], [16], [17], [22], trabajos que realizan propuestas similares en dominios diferentes.

Como trabajo futuro se aplicará este método de forma experimental en casos de estudios que hoy se están definiendo, para verificar sus resultados y detectar futuras mejoras o extensiones a realizar. En concreto, y como primer paso, este trabajo se circunscribe a la realización de una nueva versión del Sistema de Gestión de Ferias y Eventos para la Cámara de Comercio de la República de Cuba.

IV. Conclusiones

La revisión de la literatura especializada relacionada con MDA y con las principales metodologías para el desarrollo de software, permitió poder identificar los modelos necesarios para la especificación de las funcionalidades o comportamiento del sistema y su adaptación a la propuesta de MDA en cada uno de los niveles.

El método propuesto cuenta con los modelos necesarios que se corresponden con los diferentes niveles de abstracción de MDA. Para ello, se han definido dos perspectivas de análisis: la perspectiva del negocio, que coincide con los modelos de nivel CIM, y que se centra en el análisis de las características del negocio en el que se desenvolverá el sistema que se va a construir; y la perspectiva del comportamiento del sistema, que coincide con los modelos de nivel PIM y PSM de la arquitectura de MDA, y que se centra en el análisis de las funcionalidades y procesos necesarios para el desarrollo del sistema.

El principal aporte de este trabajo es la definición de un nuevo enfoque para el desarrollo de sistemas de gestión de ferias y eventos. En concreto, se ha propuesto un guía para la construcción de aplicaciones que faciliten el trabajo de las entidades que se dedican a la gestión de ferias y eventos. Este nuevo enfoque permite a los desarrolladores de software aprovechar al máximo los beneficios del paradigma de computación dirigido por modelos, favoreciendo el desarrollo de sistemas con calidad.

Como parte de la propuesta, se especifican los modelos a realizar por cada nivel de la arquitectura. Además, se está trabajando en un conjunto de tareas necesarias para garantizar mediante una herramienta informática que se pretende implementar, la generación y transformación de forma automatizada, de cada uno de los modelos propuestos a nivel CIM, PIM y PSM. De esta manera, se ha alcanzado el principal objetivo de este trabajo, que es la definición teórica de un método basado en MDA para el desarrollo de sistemas de gestión de ferias y eventos, permitiendo mejorar la productividad y calidad del proceso de desarrollo de software, así como elevar la calidad de los productos desarrollados.

V. Referencias

[1] OMG. Model Driven Architecture. Document number ormsc/2001-07-01. s.l. : Miller, J., Mukerji, J., 2001.

[2] Bocanegra, J. y Peña, J. Una Aproximación MDA para Modelar Transacciones de Negocio a Nivel CIM. En: Jornadas de Ingeniería del Software y Bases de Datos. España : s.n., 2008. págs. 82-91.

[3] Singh, Y. y Sood, M. The Impact of the Computational Independent Model for Enterprise Information System

Development. En: International Journal of Computer Applications. 2008. págs. 21-26.

[4] Mora, B y García, F. Software Generic Measurement Framework Based on MDA. En: IEEE Latin America Transactions. 2008. págs. 363-370.

[5] Martínez, Y y Cachero, C. MDD vs. Traditional Software Development: a Practitioners subjective Perspective. En Information and Software Technology. 2012. págs. 89-200.

[6] Claro Sánchez, Ana Margarita y Rodríguez Abreu, Ariel. Módulo Cuento, del producto ¨Mis Mejores Cuentos¨. La Habana : s.n., 2011.

[7] Sommerville, Ian. Ingeniería de Software. 8va Edición. Boston : Addison-Wesley, 2007. 9780321313799.

[8] Beck, Kent. Extreme Programming Explained. Massachusetts : Addison Wesley, 1999. 0201616416.

[9] Schwaber, K., Beedle, M y Martin, R.C. Agile Software Development with SCRUM. s.l. : Prentice Hall, 2001.

[10] Jacobson, Ivar, Booch, Grady y Rambaugh, James. El Proceso Unificado de Desarrollo de Software. Madrid : Addison Wesley, 2000.

[11] Villarroel González, Luis Ramiro y Montalvo Yépez , César Alejandro. Aplicación de la metodología MSF v4.0 a la definición e implementación de arquitecturas orientadas a objetos en visual studio .net 2005, caso práctico g5 sharing files. Sangolqui : s.n., 2008.

[12] Molina Moreno, Pedro Juan. Especificación de interfaz de usuario: de los requisitos a la generación automática. Valencia : s.n., 2003.

[13] Pressman, Roger S. Ingeniería del Software: Un Enfoque Práctico. Sexta Edición. 2005. pág. 900.

[14] De Castro, V., Marcos, E. y López Sanz, M. A Model Driven Method for Service Composition Modeling: A Case Study. En: International Journal of Web Engineering and Technology. s.l. : Inderscience Enterprise Ltd, 2006. págs. 335-353. 1476-1289.

[15] V. de Castro, María. Aproximación MDA para el desarrollo orientadp a servicios de sistemas de información web: del Modelo de Negocio al Modelo de Composición de Servicios Web. Madrid : s.n., 2007.

[16] Vara, J. M., de Castro, V. y Marcos, E. WSDL automatic generation from UML models in a MDA framework. En: International Conference on Next. [ed.] S.Yong Han, D. Hung-Chang Du, M. Paprzcycki A. Abraham. IEEE Computer Society Press. 2005. págs. 319-324.

[17] Silega Martínez, Nemury, y otros, y otros. Framework basado en MDA y ontologías para la representación y validación de modelos de componentes. En: Revista Cubana de Ciencias Informáticas. La Habana : Ediciones Futuro, 2014. 1994-1536.

[18] Cuesta, M. A. y López, T. M. Comparativo de herramientas MDA (AndroMDA, ArcStyler, Op timalJ) Vector. 2009. págs. 50-58. Vol. 4.

[19] Espinosa, M. M. Buenas prácticas sobre gestión de configuración en proyectos con metodología MDA. En: Revista Cubana de Ingeniería. 2012. págs. 43-49.

[20] Haitao, W. y Wei, C. Research on Modeling and Model Converting Approaches Based on MDA. En: Second WRI World Congress on Software Engineering. s.l. : IEEE, 2010.

[21] Kardoš, M. y Drozdová, M. Analytical Method of CIM to PIM Transformation in Model Driven Architecture (MDA) JIOS. 2010. págs. 89-99. Vol. 34. 1.

[22] Martínez, Y. y Cachero, C. Evidencia empírica sobre mejoras en productividad y calidad en enfoques MDD: un mapeo sistemático. En: REICIS Revista Española de Innovación, Calidad e Ingeniería del Software. 2011. págs. 6-27. Vol. 2. 7.