ESTADO DEL ARTE EN EL DESARROLLO DE APLICACIONES WEB BASADO EN INGENIERÍA DIRIGIDA POR MODELOS.

STATE OF THE ART IN THE DEVELOPMENT OF WEB APPLICATIONS BASED ON MODEL DRIVEN ENGINEERING.

Alexis Cabrera Mondeja, MSc.
Máster en Nuevas Tecnologías para la Educación (Cuba).
Ingeniero Químico.
alexiscm72@hotmail.com

Oiner Gómez Baryolo, PhD.
Doctor en Ciencias Informáticas (Cuba).
Decano de la Facultad de Sistemas y Telecomunicaciones en la
Universidad Tecnológica Ecotec, Ecuador.
ogomez@ecotec.edu.ec

ARTÍCULO DE REVISIÓN BIBLIOGRÁFICA
Recibido: 3 de noviembre de 2016.
Aceptado: 23 de diciembre de 2016.

RESUMEN

Un enfoque que ha tomado auge en los últimos años es el uso de técnicas de inegeniería dirigida por modelos (*MDE, Model Driven Engineering*) para la actualización de software, éstas técnicas no se usan solamente para crear nuevos sistemas, sino también para modernizar o evolucionar el software existente. Un enfoque muy utilizado con este fin es la ingeniería web dirigida por modelos (*MDWE, Model Driven Web Engineering*) que permite aplicar los principios de MDE en el desarrollo de aplicaciones web. A pesar de que algunos enfoques ya usan MDE para el desarrollo de aplicaciones web, todavía falta mucho por avanzar. Existen muchas limitaciones aún, ya que muchas propuestas están atadas a tecnologías y estilos arquitectónicos específicos, limitando la creación parametrizable de aplicaciones usando tecnologías diferentes, generalemente solo crean aplicaciones cliente-servidor y usan tecnologías específicas como PHP, JSP, Python, etc. Esto limita la generación de aplicaciones a partir de modelos, así como la movilidad de las mismas. El presente artículo propone hacer un estudio de los enfoques más usados en la ingeniería dirigida

por modelos, profundizando en aquellos que se centran en el desarrollo de aplicaciones web.
Palabras clave: ingeniería dirigida por modelos, ingeniería web basada en modelos, ingeniería inversa, metamodelos, lenguajes de dominio específico.

ABSTRACT

One approach that has evolved in recent years is the use of MDE (Model Driven Engineering) techniques for software upgrading, these techniques are not only used to create new systems, but also to modernize or evolve existing software. One approach widely used for this purpose is Model Driven Web Engineering (MDWE), which allows the application of the principles of MDE in the development of web applications. Although some approaches already use model-driven engineering for web application development, there is still a long way to go. There are many limitations, since many proposals are tied to specific architectural technologies and styles, limiting the parameterizable creation of applications using different technologies, generally only create client-server applications and use specific technologies like PHP, JSP, Python, etc. This limits the generation of applications from models, as well as the mobility of them. This article proposes to make a study of the most used approaches based on Model Driven Engineering, deepening in those that focus on the development of web applications.
Keywords: model driven engineering, model driven web engineering, reverse engineering, metamodels, domain specific languages

INTRODUCCIÓN

El desarrollo de software ha ido evolucionando a través de los años, a pesar de las diversas investigaciones y esfuerzos que se realizan en función de facilitar el desarrollo y evolución de los mismos, todavía la mayoría de los sistemas se producen basado en un modelado que solo es utilizado en el análisis y diseño. Una vez concluido el desarrollo, los modelos son desechados, siendo UML (*Unified Modeling Language*) un estándar muy utilizado con este fin.

Existen numerosos enfoques de desarrollo de software que se centran en modelos. Los más comunes son MDE (Model Driven Engineering) y MDD (Model Driven Development), existen además los enfoques MDA (Model Driven Architecture), MDSD (Model Driven Software Development), MDWD (Model Driven Web Development), MDWE (Model Driven Web Engineering), Model Driven Reverse Engineering (MDRE), entre otros, casi todos son similares y se enfocan en diferentes aspectos arquitectónicos o de plataforma.

REVISIÓN TEÓRICA

Ingeniería dirigida por modelos.

La ingeniería dirigida por modelos (*MDE*) y el desarrollo dirigido por modelos (MDD) son genéricos y describen un enfoque para representar a los sistemas como modelos que se ajustan a metamodelos. A través de procesos de transformación de modelos se pueden obtener varias representaciones. (Liddle, 2011). MDD, un enfoque de OMG (*Object Management Group*), considera que idealmente los modelos son independientes de la plataforma (*PIM, Platform Independent Model*), su derivación convierte modelos PIM a modelos específicos de la plataforma (*PSM, Platform Specific Model*).

Según plantea (Trujillo, 2007) en MDD los modelos son escritos en un lenguaje de dominio específico (*DSL, Domain Specific Language*) que específica detalles del diseño de un programa. A partir de las transformaciones realizadas a uno o varios modelos se puede obtener un modelo, centrándose el desarrollo en la transformación de modelos a ejecutables.

El desarrollo de software es un proceso complejo, cada vez las reglas de negocio y los requerimientos son más complicados, aumentan los riesgos de fallas y los tiempos de desarrollo. MDE intenta mejorar la productividad del programador, la reusabailidad y la mantenibilidad de los sistemas. Las técnicas de la ingeniería dirgida por modelos ayudan, entre otras cosas, a detectar errores tempranamente tales como errores de diseño, omisiones o problemas de comunicación con los clientes. (Liddle, 2011)

El enfoque MDE coloca a los modelos en el centro del proceso de desarrollo, "*everything is a model*" (Bézivin, 2004) y se centra en los modelos como un proceso de evolución del software a través de su ciclo de vida, así como evoluciona el código, los modelos evolucionan a través del tiempo, siendo éstos actualizados a medida que el código se actualiza. Uno de los objetivos principales del desarrollo moderno es la reutilización de código con vistas a optimizar tiempo, la reutilización de modelos durante el ciclo de vida del software contribuye también a la optimización de tiempo muchas veces generando parte del código de la aplicación. (David, 2013)

Uno de los principales objetivos del enfoque MDE es mejorar en lo mayormente posible el proceso de desarrollo al capturar características claves de un sistema en modelos que son sincronizados, combinados y transformados en diferentes niveles de abstracción. A diferencia del modelado tradicional, los modelos no solo forman parte de la documentación, éstos son procesados y transformados durante la evolución del sistema. (Voelter, 2007). MDE considera, entre sus más importantes enfoques, el

ISSN: 1390-9320, Vol. 4, No. 1, febrero 2017

modelado de cada uno de los aspectos del sistema, a través de lenguajes de dominio específicos. Los DSL se enfocan en áreas muy específicas permitiendo que el modelado se centre en cada uno de los procesos del negocio. (Jouault, 2010)

El desarrollo centrado en modelos promete un alto nivel de automatización, los modelos escritos en DSL pueden describir muy claramente cada uno de los procesos del negocio. Cada modelo captura información y un modelo puede derivarse de otro modelo a través de transformaciones, finalmente el desarrollo se concentra en un proceso de transformación de modelos a ejecutables, considerados también modelos.

MDE centra a los modelos como artefactos de primera clase para mantener, analizar, simular y transformarlos en código o en otros modelos. El metamodelado es el concepto clave del paradigma MDE, y se pretende como la manera de dotar un lenguaje con una notación abstracta, separando la sintaxis abstracta y la semántica del lenguaje de sus notaciones concretas diferentes. (Gargantini, 2010)

Los modelos deben ser formales, donde cada modelo es una instancia de un metamodelo. El metamodelo define el vocabulario y la gramática usada para crear los modelos. Los DSL se usan para crear modelos a través de la definición de un metamodelo y usan una sintaxis concreta para representarlos. La sintaxis concreta puede ser textual, gráfica u otros objetos como tables, árboles, etc., y debe representar fielmente los conceptos que intenta describir el DSL. (Voelter, 2007)

A partir del modelado del negocio usando varios DSL que expresen cada uno de los procesos, el paradigma MDE permite transformar estos modelos en otros modelos e incluso en modelos ejecutables. Es muy difícil en la práctica obtener una aplicación completamente funcional a través de la generación de código a partir de los modelos de un DSL, pero contribuiría notablemente a agilizar el proceso de desarrollo, y a la reutilización de los modelos en el ciclo de vida del software.

Un enfoque, común en varias investigaciones, defiende la creación de modelos que contengan exclusivamente conceptos del dominio. Tales lenguajes son llamados lenguajes de modelado de dominio específico (*DSML, Domain-specific Modelling Languages*). DSML es el equivalente de DSL, y se encarga de representar eficientemente los conceptos de un dominio particular. (Montrieux, 2013) Este enfoque usa un lenguaje de modelado personalizado para representar a un sistema en un dominio particular. Las herramientas de DSML permiten crear modelos específicos de un dominio dado y generar código usando modelos y conceptos específicos según las necesidades del cliente. (Liddle, 2011)

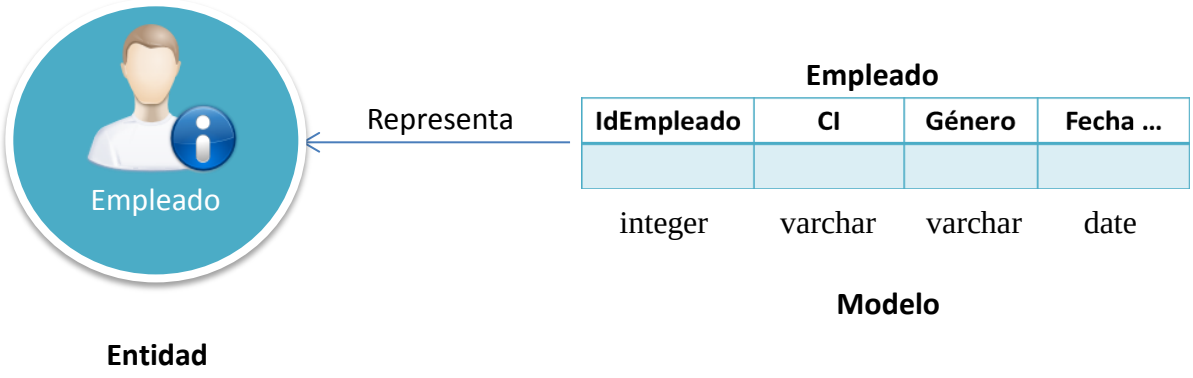
Los principios básicos de MDE están basados en dos conceptos: sistema y modelo, y dos relaciones básicas: conformidad y representación. A partir de esto se puede dar la

ISSN: 1390-9320, Vol. 4, No. 1, febrero 2017

primera definición de modelo en el contexto de la ingeniería dirigida por modelos. (Bézivin, Model driven engineering: An emerging technical space. , 2006)

Los modelos se usan comúnmente para representar situaciones de la vida real, pueden representar sistemas, entidades, etc.

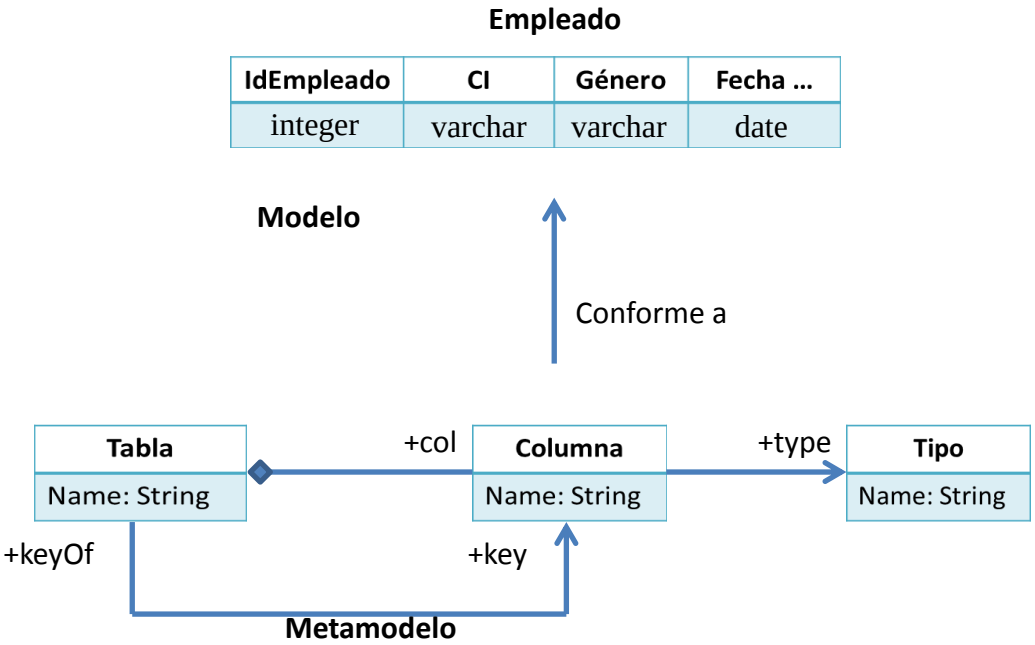
Figura 1: Un modelo representa a una entidad empleado.



Fuente: elaboración propia.

El modelo que representa la entidad empleado es descrito a través un metamodelo que describe sus elementos y la forma en que interactúan.

Figura 2: Un metamodelo describe el modelo de la entidad empleado.



Fuente: elaboración propia.

ISSN: 1390-9320, Vol. 4, No. 1, febrero 2017

El enfoque MDA.

Uno de las más conocidas iniciativas de desarrollo de software basado en modelos es *Model Driven Architecture (MDA*), un enfoque desarrollado y mantenido por el consorcio *Object Management Group (OMG*).

MDA plantea como uno de sus principales objetivos separar la lógica de la aplicación y del negocio, de la tecnología de la plataforma destino, permitiendo que los cambios en la plataforma subyacente no afecten las aplicaciones existentes, y la lógica del negocio pueda evolucionar independientemente de la tecnología empleada. Las herramientas basadas en MDA deben permitir la creación de modelos, tanto de la aplicación como de la lógica del negocio, y generar código para diferentes plataformas de destino a través de la transformación de modelos. Este enfoque incrementa notablemente el nivel de abstracción en el proceso de desarrollo. En lugar de escribir código para una plataforma específica en un lenguaje de alto nivel, el proceso de desarrollo se centra en la creación de modelos específicos para el dominio de la aplicación e independientes de la plataforma. Los estándares de OMG para este enfoque son UML, MOF (Meta-Object Facility), XMI (XML Metadata Interchange), y MOF/QVT (Query/View/Transformations), entre otros. (Moreno N. R., 2008)

MDA planteaba, en sus inicios, entre sus principios fundamentales que un modelo independiente de la plataforma (PIM) podría ser expresado en UML y que, a través de la supuesta estabilidad de las versiones de UML en el tiempo, los recursos correspondientes podrían ser preservados durante largos períodos de tiempo. Los medios concretos para generar los modelos específicos de la plataforma a partir de PIM no estaban claramente establecidos en el momento ya que el número de este tipo de plataformas de destino era bastante limitado (principalmente CORBA, J2EE / EJB y DotNet). (Trujillo, 2007). El alcance de estas plataformas de destino luego se amplió, la noción de modelos (incluyendo PIM y PSM) se amplió más allá de simples modelos UML, y se sugirió automatizar la generación de los PSM a partir de PIM mediante transformaciones de modelos utilizando el estándar *Query View Transformation* (QVT). (Liddle, 2011)

Aunque MDA no es el único enfoque de la ingeniería dirigida por modelos, numerosas iniciativas han aportado al desarrollo y evolución de la ingeniería dirigida por modelos. MDA y DSL son soluciones que se encuentran más relacionadas actualmente, MDA ofrece a los DSL la idea de usar una colección de metamodelos para capturar las fases de un sistema en desarrollo o en mantenimiento. En tanto los DSL brindan a MDA el enfoque de que un único lenguaje de propósito general, aún UML, no es capaz

ISSN: 1390-9320, Vol. 4, No. 1, febrero 2017

de capturar todas las necesidades de los diseñadores, administradores y usuarios de un sistema dado. (Groenewegen, 2013)

MDE aporta la noción de múltiples dimensiones de modelado y un proceso de ingeniería de software para MDA. MDE tiene como propósito aumentar la rentabilidad que se deriva del esfuerzo en desarrollo de software de una empresa. En muchos casos cuando se habla de MDA o MDE la única meta asociada con estos términos es reducir la sensibilidad de los artefactos de software para el cambio en las plataformas de despliegue utilizando un PIM y un PSM. (Adamkó, 2010)

El framework MDA está basado principalmente en los modelos PIM y PSM, y en la transformación de modelos entre ellos. Estos conceptos del dominio muestran un determinado grado de independencia de diferentes plataformas (.NET, and J2EE). El sistema puede ser compilado usando cualquiera de estas plataformas de destino mediante la transformación del PIM a un modelo de plataforma específico (PSM). A través de los PSM se detalla cómo el sistema utiliza un tipo particular de plataforma. Finalmente, el código de la aplicación se considera una forma de PSM (en el nivel más bajo).

En MDA, una plataforma es un conjunto de subsistemas y tecnologías que ofrece un conjunto de funcionalidades a través de interfaces y patrones de uso específicos, que cualquier aplicación, soportada por la misma, puede utilizar sin preocuparse por los detalles de cómo se implementa la funcionalidad (Miller, 2003). Al igual que en MDSD (Model Driven Software Development), cada modelo en MDA se ajusta a un metamodelo, que en MDA puede definirse utilizando MOF.

En MDA las transformaciones son el núcleo principal del enfoque. Convertir un modelo a otro modelo del mismo sistema, son transformaciones que se pueden llevar a cabo siguiendo muchas vías: usando tipos, marcas, plantillas, etc. De esta manera el desarrollo de software se convierte en un proceso iterativo de transformaciones de modelos: cada paso transforma un PIM del sistema a un nivel dentro de un PSM al siguiente nivel, hasta que que se alcanza la implementación final, con la particularidad que cada PSM de una transformación puede convertirse en el PIM de la siguiente transformación (con otro nivel de abstracción). En este contexto, la implementación es justo otro modelo, el cual provee toda la información necesaria para construir el sistema y ponerlo en funcionamiento. (Moreno N. R., 2008)

La ingeniería inversa dirigida por modelos.

La ingeniería inversa dirigida por modelos (MDRE, *Model-Driven Reverse Engineering*) se define comúnmente como la aplicación de los principios de MDE a la ingeniería inversa. Es muy común aplicar estas técnicas en la migración y actualización de

ISSN: 1390-9320, Vol. 4, No. 1, febrero 2017

sistemas hacia versiones más modernas, donde MDE intenta obtener modelos a partir de los sistemas anteriores para generar los nuevos sistemas (Bruneliere, 2014).

La aplicación de MDE en ingeniería inversa es muy reciente (S. Rugaber, 2004). En sus principios, los modelos eran usados para especificaciones de sistemas antes de sus implementaciones. Sin embargo, MDRE propone construir y usar modelos de la implementación del sistema, beneficiándose directamente de estas vistas de alto nivel del sistema, como modelos de diseño, o de sus modelos de dominio, con el fin de mejorar los procesos de mantenimiento y evolución del sistema.

Debido al creciente interés que ha tenido el enfoque MDRE, OMG lanzó el grupo de trabajo *Architecture Driven Modernization* (ADM), con el objetivo de proponer un grupo de metamodelos estándares para la modernización de software, como migraciones de viejas tecnologías a otras más recientes. Basado en esto, varias propuestas tales como las planteadas por (Favre, 2010), combinan estos estándares en un framework metodológico.

MDRE es útil también para fines analíticos, por ejemplo, la plataforma Moose (Girba, 2010). También es común el requerimiento de MDRE en la evolución de software basado en modelos, por ejemplo, en escenarios que incluya posibles modificaciones de sistemas obsoletos (de tipo estructural, funcional, mantenimiento, etc.) y no solo pura modernización.

En todos los casos, el uso de MDRE se hace necesario para obtener modelos de requerimientos de los sistemas para su análisis, modificación o evolución.

Los modelos son los principales artefactos en el desarrollo de software, como hemos mencionado, los modelos pueden ser usados para representar elementos del diseño y desarrollo del software durante su ciclo de vida. Los modelos independientes de la plataforma (PIM), y los modelos específicos de la plataforma (PSM), son el centro de la ingeniería avanzada y de la ingeniería inversa. En la primera, los modelos PIM son desarrollados por humanos como parte de los esfuerzos en el diseño de software. En la ingeniería inversa, estos modelos son típicamente derivados de manera automática usando transformaciones de modelos. En cualquier caso, los elementos que constituyen un modelo independiente de la plataforma tienen que ser entendidos. Por lo tanto, debemos comenzar con detalles sobre lo que constituye modelos independientes de la plataforma y cómo construirlos (Akkiraju, 2012).

Ingeniería web dirigida por modelos.

El enfoque MDWE (MDWE, *Model-Driven Web Engineering*) se encuentra en un proceso de maduración, las aplicaciones web han evolucionado hacia arquitecturas más complejas y numerosos frameworks basados en Python (Django), Ruby (Ruby on

Rails), entre otros, utilizan los principios de MDE para la generación de la aplicación a partir de modelos. Por otro lado, se han estrechado las diferencias entre las aplicaciones tradicionales y las aplicaciones web, conllevando a que los investigadores creen numeroros enfoques de MDWE. (Wimmer, 2007) identifican cinco principales grupos de métodos MDWE:

- Enfoques orientados a datos tales como RMM, WebML, y Hera tienen sus orígenes en sistemas de bases de datos, y se enfocan en aplicaciones web de datos intensivos.
- Métodos orientados a hipertexto tales como HDM, HDM-lite, WSDM y W2000 se originan de diseños hipermedia, y manejan muy bien la naturaleza del hipertexto en aplicaciones web.
- Enfoques orientados a objeto siguen el modelado tradicional de OO, e incluyen tales métodos como OOHDM, UWE, OOWS, y OO-H.
- Los métodos orientados al software toman un enfoque similar al desarrollo de software tradicional. *Web Application Extension* (*WAE*) y su extensión WAE2 ejemplifica este enfoque.
- Los métodos orientados a MDE toman explícitamente un enfoque dirigido por modelos para el desarrollo de aplicaciones web y enfatiza en la generación automática de código desde modelos de aplicaciones web.

Paralelamente a la evolución y maduración de los métodos MDWE, los investigadores han expresado su creciente preocupación por los problemas a nivel macro ya que se analiza la complejidad y dificultad que los nuevos métodos pueden generar, algunos consideran que los métodos MDWE se encuentran en una crisis similar al paradigma OO en los años 1990. (Liddle, 2011)

Los enfoques más recientes basados en MDWE responden a la creciente demanda de rigurosas prácticas de desarrollo, las aplicaciones web han evolucionado considerablemente, pero aún la administración del mantenimiento y la evolución de estas aplicaciones sigue siendo un problema. (Cicchetti, 2013)

En las últimas décadas se ha propuesto un gran número de lenguajes de modelado y herramientas de soporte para desarrollar aplicaciones web. El proceso de evolución de éstas es complejo, así como evolucionan los lenguajes y los navegadores, deben evolucionar las aplicaciones, siendo un gran reto el mantenimiento y evolución durante su ciclo de vida. (Cicchetti, 2013)

La mayoría de los enfoques existentes proporcionan los medios para desarrollar aplicaciones web complejas por medio de modelos que son considerados una parte

ISSN: 1390-9320, Vol. 4, No. 1, febrero 2017

integral del proceso de desarrollo. Estos definen aspectos relevantes de una APW (Aplicación Web) y sus funcionalidades de acuerdo a la semántica del lenguaje de modelado utilizado; las transformaciones de modelos pueden generar los artefactos para una plataforma determinada, considerándose estas transformaciones como aplicaciones funcionales que generan partes del sistema.

A través del proceso de desarrollo los modelos pueden cambiar o evolucionar y los sistemas pueden ser generados o regenerados, sin embargo, es un proceso complejo ya que los modelos son representaciones abstractas (Fowler, 2005) y varios activos no se reflejan directamente en los modelos, como son los contenidos almacenados en back-ends persistentes y los artefactos obtenidos a través de personalizaciones manuales.

La modificación de modelos, a través del ciclo de vida, podría requerir adaptaciones adicionales para mantener la validez del sistema. Procesos como la migración de datos de un sistema antiguo a uno nuevo puede ser impulsada por una relación de conformidad bien definida entre los datos y su esquema, en tanto las correspondencias entre las plantillas de diseño personalizado y el modelo son menos complicadas. (Cicchetti, 2013)

La ingeniería de las APW requiere de la atención en muchos aspectos como son el modelado de datos, la interfaz de usuario, las acciones, la validación de datos y el control de acceso. En el desarrollo de APW estos aspectos están soportados por lenguajes débilmente acoplados que requieren abundante código repetitivo y carecen de verificación estática. Un gran reto para los DSL en el dominio web es desarrollar un lenguaje conciso de alto nivel, declarativo para la definición de APW, donde los requerimientos sean soportados por sub-lenguajes especializados, integrados y que puedan derivar implementaciones automáticamente.

Se requiere investigar y crear DSL adecuados para cada subdominio de la aplicación, así como una integración de éstos que aseguren la consistencia de los modelos en cada uno de los dominios. El proceso de investigación es relevante para descubrir buenas abstracciones para el dominio de la ingeniería web, así como para el desarrollo de familias de DSL.

El núcleo de las APW es su modelo de datos, la APW debe organizarse para preservar la consistencia de datos con respecto al modelo de datos durante las actualizaciones, eliminaciones, e inserciones. Las propiedades del núcleo de un modelo de datos están formadas por restricciones estructurales, es decir, los miembros de datos de las relaciones entre entidades. Algunas propiedades de consistencia no pueden ser expresadas como restricciones estructurales. Además, algunas restricciones de

ISSN: 1390-9320, Vol. 4, No. 1, febrero 2017

integridad de datos no pertenecen directamente a los datos persistentes. (Groenewegen, 2013)

El dominio técnico de la ingeniería web ha sido algunas veces reconocido como adecuado para el enfoque de desarrollo de software dirigido por modelos. La combinación de los dos ha sido nombrada *Model Driven Web Development* (MDWD). Las metodologías actuales enfrentan un gran reto, muchas de ellas se centran en el modelado para tecnologías específicas y no en modelos independientes de la plataforma. Muchas de estas metodologías MDWD usan modelado gráfico. Otras usan modelado textual, un ejemplo de enfoque textual es WebDSL, un lenguaje de dominio específico cuyo ámbito abarca las aplicaciones web interactivas con un modelo de datos amplio. (Groenewegen, 2013)

Ambos enfoques están en el ámbito del dominio de las aplicaciones web usando abstracción y automatización. Sin embargo, tienen poco en común en sus implementaciones y la evolución de estos difiere bastante. En comparación con la cambiabilidad de los dos enfoques antes mencionados el enfoque gráfico es probable el que pierda, ya que se centra en una mayor cantidad de aspectos, acomoda refinamientos manuales en varios niveles de abstracción e introduce altos costos y reduce la flexibilidad. (VAN DIJK, 2009)

Algunas propuestas recientes (Meta Programming System, s.f.) han encontrado la necesidad de modelar funcionalidades, y debido a esto se han extendido los modelos con algunos comportamientos operacionales simples. Otro factor influyente es el aumento de la complejidad de las interfaces web que ha provocado incentivar esfuerzos en la investigación dirigidos a metodologías de desarrollo centradas en la interface y sus mecanismos de comunicación con la lógica de la aplicación. Tal es el caso de UIML.

Según (Gómez, 2011) los enfoques MDWE existentes pueden ser extendidos con éxito a través de nuevos modelos que reúnan específicamente características relacionadas. El autor se basa en tres suposiciones: el comportamiento de una aplicación web no debe restringirse a unas simples operaciones de actualización, y por tanto debe ser dirigido de una manera rigurosa, la información del contexto y el comportamiento del dominio dentro de la aplicación web deberían ser dirigidos con métodos OO ya probados, como en cualquier otra aplicación distribuida, y finalmente a pesar de que puedan tener la misma capa lógica subyacente, la apariencia y el comportamiento de la interfaz difieren en gran medida entre las aplicaciones web y las tradicionales.

El proceso de evaluación de la calidad de las APW requiere de comprender en detalles sus diferencias con las aplicaciones convencionales en cuanto a requerimientos de

ISSN: 1390-9320, Vol. 4, No. 1, febrero 2017

usuario, expectativas y comportamiento. Durante los últimos años, diversas investigaciones han estudiado las diferencias entre las APW y las convencionales, además se han actualizado los estándares ISO 25010/25012 con propuestas de extensiones de los modelos de especificaciones y requerimientos para APW (Lew, 2012).

Muchas veces el enfoque MDE ha sido asociado con el desarrollo ágil, precisamente dentro de sus objetivos está proporcionar herramientas que permitan optimizar tiempos de desarrollo. Los métodos ágiles son particularmente atractivos para las APW dada su rápida evolución y cortos ciclos de vida, entre otros factores. Generalmente las aplicaciones web se desarrollan incrementalmente con fuerte retroalimentación con los expertos del dominio para validar diferentes prototipos en ejecución.

Según (Luna, 2009) desafortunadamente los enfoques más sólidos de MDWE, entre los que se encuentran a favor del desarrollo incremental e iterativo, usan un estilo de desarrollo más formal y en cascada. Métodos tales como UWE, WebML, OOWS, OO-H or OOHDM (Wimmer, 2007) definen una serie de modelos abstractos tales como contenido (llamado también datos o aplicación), navegación y modelo de presentación, que permiten generar aplicaciones a través de transformaciones automáticas de modelos, que se consideran libres de errores. El autor considera que es un enfoque atractivo ya que eleva el nivel de abstracción durante el proceso de desarrollo, lo que conlleva a que los desarrolladores se centren en modelos conceptuales y no en el código. Existe una creciente disponibilidad de técnicas que añaden sinergia a este enfoque (Luna, 2009).

En el área de ingeniería web, los esfuerzos para integrar estilos ágiles y dirigidos por modelos en el desarrollo están evolucionando, y muchos métodos carecen de una clara heurística de cómo mejorar el ciclo de vida del desarrollo con la incorporación de nuevas ideas.

Las metodologías ágiles prometen interacción constante y temprana con los clientes para comprobar que el software creado cumpla sus requerimientos, a través de la constante entrega de prototipos desarrollados en cortos periodos de tiempo. Las metodologías de MDWE facilitan la portabilidad de las especificaciones de software, la abstracción y la productividad, pero fallan en proveer interacción ágil con los clientes porque los resultados concretos se obtienen demasiado tarde. Por otra parte, mientras esta característica es suminstrada claramente por las metodologías ágiles, están fuertemente basadas en implementaciones directas y por tanto fallan al proveer abstracción, portabilidad y productividad a través de la generación automática de código. (Rivero, 2011)

ISSN: 1390-9320, Vol. 4, No. 1, febrero 2017

La mayoría de los enfoques de desarrollo de aplicaciones web se ajustan al paradigma MDD, ya que dirigen el proceso de desarrollo a través del uso de modelos que describen cada uno de los aspectos de la aplicación. Estos métodos proveen compiladores de modelos para generar implementaciones automáticas de sistemas a partir de modelos de alto nivel. Ha sido común que los enfoques de ingeniería web se centren en el contenido, navegación y presentación como aspectos más relevantes, sin embargo, estas propuestas también presentan algunas limitaciones, especialmente cuando se trata de modelar otros aspectos, tales como estilos de arquitectura o distribución. Además, las complejidades de las nuevas aplicaciones web deben dirigir la atención hacia aspectos del dominio y del modelo de datos. La complejidad y los requerimientos en las aplicaciones web están creciendo constantemente, mientras que las tecnologías soportadas y plataformas evolucionan rápidamente. (Moreno N. R., 2008)

La evolución de las APW a través de las técnicas de MDE implica según (Moreno, 2008): (a) la definición de nuevos modelos y elementos de modelado que capturen requerimientos adicionales; (b) la redefinición del metamodelo para la manipulación de estas características adicionales; (c) la adaptación del proceso de desarrollo para incorporar el nuevo aspecto y la información que representa; (d) la adaptación del proceso de modelado y las herramientas de generación de código que soporten el método.

Los actuales sistemas web necesitan interoperar con otras aplicaciones externas, algo que requiere su integración con servicios web de terceros, portales, y también con sistemas anteriores. Finalmente, muchas de estas propuestas de ingeniería web no explotan completamente todos los potenciales beneficios de MDSD, tales como independencia de la plataforma, trasformación y mezcla de modelos, metamodelos. (Moreno, 2008)

El enfoque MDA cubre numerosos aspectos y cuestiones (MOF-metamodelos, perfiles UML, transformaciones de modelos, lenguajes de modelado y herramientas, etc.) prometiendo la interoperabilidad necesaria entre los modelos y las herramientas de los proveedores de manera independiente. (Moreno N. R., 2008)

Las aplicaciones web han evolucionado y los enfoques de MDWE han evolucionado apara ajustarse a los cada vez más complejos escenarios en el desarrollo de APW, cada vez son más los dominios que abarcan y migran de desktop imponiendo nuevos requerimientos.

A pesar de que algunos enfoques ya usan la ingeniería dirigida por modelos para el desarrollo de aplicaciones web, todavía falta mucho por avanzar, entre los enfoques más conocidos están WebML (*Web Modeling Language*), UWE (*UML-based Web*

ISSN: 1390-9320, Vol. 4, No. 1, febrero 2017

Engineering), XIS2 (*eXtreme Modeling Interactive Systems*), the *OutSystems Agile Platform*, OOHDM *(Object-Oriented Hypermedia Design Model*), or OPM/Web (*Object Process Methodology for Developing Web-Applications*) (Wimmer, 2007). Existen algunos enfoques, que, aunque no están orientados al desarrollo de aplicaciones web, manejan aspectos tales como el uso de meta-metamodelos, modelado de interfaz de usuario, o el uso de herramientas de prototipado para proveer una vía para que los expertos del negocio dibujen y comuniquen ideas. (da Silva, 2010)

Existen muchas limitaciones aún ya que muchas propuestas están atadas a tecnologías y estilos arquitectónicos específicos, limitando la creación parametrizable de aplicaciones usando tecnologías diferentes, generalemente solo crean aplicaciones cliente-servidor y usan tecnologías específicas como PHP, JSP, Python, etc. Esto limita la generación de aplicaciones a partir de modelos, así como la movilidad de las mismas. La mayoría de estas propuestas fueron originalmente pensadas para determinados tipos de aplicaciones web (sistemas de información, aplicaciones hipermedia, etc.), por lo que se ajustan a un grupo determinado de aspectos (navegación, presentación, etc.). Aunque son muy buenas para modelar aspectos específicos limitan su extensión a otros dominios como el modelado de procesos, entre otros.

CONCLUSIONES

Existen varios enfoques de MDE para el modelado de software, obtener un enfoque que resuelva todas las limitaciones actualmente existentes es bien complejo.

Existen muchas propuestas que adicionan demasiadas características llegando a ser demasiado complejas y frágiles.

Muchas propuestas están atadas a tecnologías y estilos arquitectónicos específicos, limitando la creación parametrizable de aplicaciones usando tecnologías diferentes, generalemente solo crean aplicaciones cliente-servidor y usan tecnologías específicas como plataformas basadas en Ruby on Rails, Django, etc.

REFERENCIAS BIBLIOGRÁFICAS

Adamkó, A. &. (2010). Developing Web-Based Applications Using Model Driven Architecture and Domain Specific Languages. *In Proceedings of the 8th International Conference on Applied Informatics*, 287-293.

Akkiraju, R. M. (2012). Reverse engineering platform independent models from business software applications. *INTECH Open Access Publisher*.

Bézivin, J. (2004). In search of a basic principle for model driven engineering. . *Novatica Journal, Special Issue, 5(2)*, 21-24.

Bézivin, J. (2006). Model driven engineering: An emerging technical space. . *In Generative and transformational techniques in software engineering Springer Berlin Heidelberg.*, 36-64.

Bruneliere, H. C. (2014). Modisco: A model driven reverse engineering framework. . *Information and Software Technology, 56(8)*, 1012-1032.

Cicchetti, A. D. (2013). Managing the evolution of data-intensive web applications by model-driven techniques. *Software & Systems Modeling, 12(1)*, 53-83.

da Silva, A. R. (2010). A reference model for the analysis and comparison of MDE approaches for web-application development. *Journal of Software Engineering and Applications, 3(05)*, 419.

David, E. R. (2013). Ingeniería dirigida por Modelos. *ALGHORITMIC, 6*.

Favre, L. (2010). Model Driven Architecture for Reverse Engine ering Technologies: Strategic Directions and System Evolution, IGI Global - . *Premier Reference Source*.

Fowler, M. (2005). *Language Workbenches: The Killer-App for Domain Specific Languages?* Obtenido de http://martinfowler.com: http://martinfowler.com/articles/languageWorkbench.html

Gargantini, A. R. (2010). Combining formal methods and mde techniques for model-driven system design and analysis. *INTERNATIONAL JOURNAL, 1*.

Girba. (2010). The Moose Book. Self Published.

Gómez, J. C. (2011). On Conceptual Modeling of Device-Independent Web Applications: Towards a Web-Engineering Approach. . *Ieee multimedia, 8(2)*, 26-39.

Groenewegen, D. M. (2013). Integration of data validation and user interface concerns in a DSL for web applications. . *Software & Systems Modeling, 12(1)*, 35-52.

Groenewegen, D. M. (2013). Integration of data validation and user interface concerns in a DSL for web applications. . *Software & Systems Modeling, 12(1)*, 35-52.

Jouault, F. V. (2010). Inter-DSL coordination support by combining megamodeling and model weaving. *In Proceedings of the 2010 ACM Symposium on Applied Computing. ACM*, 2011-2018.

Lew, P. O. (2012). An integrated strategy to systematically understand and manage quality in use for web applications. *Requirements Engineering, 17(4)*, 299-330.

Liddle, S. W. (2011). Model-driven software development. . *Handbook of Conceptual Modeling. Springer Berlin Heidelberg*, 17-54).

Luna, E. R. (2009). Bridging test and model-driven approaches in web engineering. *International Conference on Web Engineering. Springer Berlin Heidelberg*, 136-150.

Meta Programming System. (s.f.). Obtenido de http://www.jetbrains.com: http://www.jetbrains.com/mps

Miller, J. a. (2003). The MDA Guide. *Draft v. 2.0, OMG doc.*, 01-03.

Molina, F. &. (2009). Integrating usability requirements that can be evaluated in design time into Model Driven Engineering of Web Information Systems. *Advances in Engineering Software, 40(12)*, 1306-1317.

Montrieux, L. Y. (2013). Issues in representing domain-specific concerns in model-driven engineering. *In Modeling in Software Engineering (MiSE), 2013 5th International Workshop. IEEE*, 1-6.

Moreno, N. M. (2008). Addressing new concerns in model-driven web engineering approaches. *International Conference on Web Information Systems Engineering. Springer Berlin Heidelberg*, 426-442.

Moreno, N. R. (2008). An overview of model-driven web engineering and the mda. . *Web Engineering: Modelling and Implementing Web Applications. Springer London*, 353-382.

Rivero, J. M. (2011). Improving Agility in Model-Driven Web Engineering. . *CAiSE Forum (Vol. 734)*, 163-170.

S. Rugaber, K. S. (2004). Model driven reverse engineering, . *IEEE Software 2*, 45–53.

Trujillo, S. B. (2007). Feature oriented model driven development: A case study for portlets. *In Proceedings of the 29th international conference on Software Engineering. IEEE Computer Society*, 44-53.

UWE–UML–Based Web Engineering. (s.f.). Obtenido de UWE–UML–Based Web Engineering: http://uwe.pst.ifi.lmu.de

VAN DIJK, D. A. (2009). CHANGEABILITY IN MODEL DRIVEN WEB DEVELOPMENT. *Doctoral dissertation, University of Amsterdam*.

Voelter, M. &. (2007). Product line implementation using aspect-oriented and model-driven software development. . *In Software Product Line Conference, 2007. SPLC 2007. 11th International. IEEE*, 233-242.

Webml. (s.f.). Obtenido de Webml: http://www.webml.org

Wimmer, M. S. (2007). On the integration of web modeling languages: Preliminary results and future challenges. *Workshop on Model-driven Web Engineering (MDWE), held in conjunction with ICWE, Como, Italy.*