Cómo integrar el modelo CMMI al modelo de desarrollo de software MDD

Tesis

Rosalba Matos Mareño

Maestría en Ingeniería de Software
Facultad de Informática
Universidad Nacional de La Plata

Julio del 2017

Documento maquetado con TEXIS v.1.0.

Cómo integrar el modelo CMMI al modelo de desarrollo de software MDD

Presentada a la Facultad de Informática de la UNLP como parte de los requisitos para la obtención del título de Master en Ingeniería de Software
2017

Maestría en Ingeniería de Software
Facultad de Informática
Universidad Nacional de La Plata

Julio del 2017

ISBN

- *A todo aquel se sienta motivado luego de leer este trabajo, a profundizar más en el tema de integración de los Modelos MDD y CMMI-DEV.*
- *A mi directora Dra. Claudia Pons quien siempre de manera amable y oportuna orientó éste trabajo y me aportó su conocimiento en el tema.*
- *A mis hijos Javier e Ivan, a esposo Alvaro, y a toda mi familia por su permanente apoyo.*

Agradecimientos

Gracias!

- *A mi Directora Claudia Pons por su útil orientación con el modelo MDD.*
- *Y al Dr. Pat O'Toole por su amable contribución con sus valiosos conocimientos del modelo CMMI.*

Resumen

Integrar es hacer que una cosa se incorpore a algo para formar parte de ello. Esta acción es recomendable entre los modelos MDD, y CMMI-DEV con el fin de complementarlos y fortalecerlos, y así obtener procesos y productos de software de mejor calidad.

Los dos modelos tienen objetivos diferentes: CMMI para Desarrollo, proporciona buenas prácticas para el desarrollo y mantenimiento de productos y servicios, y MDD, busca tres objetivos principales: Mejorar la Portabilidad, la Interoperabilidad y la Reutilización del software. Sin embargo, ambos modelos buscan facilitar las operaciones del negocio.

"CMMI-DEV merece todo el respeto puesto que ha sido el producto de las contribuciones de expertos internacionales, tanto pertenecientes a la industria del software como al ámbito académico, intentando ambas partes responder a las crecientes necesidades y expectativas de los profesionales de la ingeniería del software". (1)

Por otra parte, MDA que es la propuesta de Modelo de Desarrollo al que nos referiremos en éste trabajo, es respaldado por el Object Management Group - OMG. El Object Management Group es un consorcio abierto que produce estándares para la interoperabilidad en el espacio de aplicaciones empresariales, y es el responsable del Lenguaje Unificado de Modelado (UML), fundamental para el MDD. Según lo definido por el OMG, "MDA es una forma de organizar y gestionar arquitecturas empresariales apoyados por herramientas y servicios automatizados tanto en la definición de los modelos como en las transformaciones entre los diferentes modelos". (20).

Buscando motivar a los organismos líderes y a la comunidad que aplica los modelos CMMI y MDD, a trabajar en pro de su integración, este trabajo presenta el resultado de un análisis de los procesos, actividades, roles y riesgos, involucrados en un desarrollo de software bajo del esquema de MDD, e identifica el artefacto propio de MDD que responde a su consistencia con los lineamientos en los niveles 2 y 3 del modelo CMMI. Para ello se ha consultado documentación del Instituto de Ingeniería de Software - SEI, del Object

Management Group - OMG, y publicaciones de investigadores y académicos.

A pesar de que CMMI no recomienda ninguna metodología de desarrollo, sería beneficioso para la expansión de las aplicaciones de MDD, que hubiera un acercamiento de la comunidad seguidora de CMMI al MDD o desarrollo por modelos. Para contribuir a esto, en el capítulo 5 se exponen los resultados de un análisis sobre cuáles serían los aspectos a tener en cuenta en una evaluación de los procesos que se siguen en un proyecto de desarrollo bajo MDD, para cumplir con las prácticas específicas de CMMI V1.3 nivele 2 y 3.

Fue intención de la autora presentar los resultados de manera sencilla para que los que se inician en cualquiera de los modelos MDD y CMMI vean claramente la posibilidad de que ambos modelos puedan coexistir complementándose en un mismo proyecto de desarrollo. El detalle técnico del manejo de los artefactos esta fuera del alcance de este trabajo.

Índice

Índice de figuras

Índice de Tablas

Capítulo 1

Introducción

"El desarrollo basado en modelos promete ser el primer verdadero salto generacional en el desarrollo de software desde la introducción de el compilador".
Bran Selic, IBM Rational Software.

1.1. Objetivo y organización del trabajo

El objetivo de la presente investigación es estudiar de qué forma se puede integrar la aplicación del modelo CMMI - DEV (Capability Maturity Model) al Modelo MDD (Model Driven Development - MDD) en un proceso de desarrollo de software. Para ello se determinará qué actividades deberían hacerse dentro de un proceso de desarrollo MDD, con el fin de cumplir los requerimientos de CMMI-DEV.

Para lograr el objetivo del trabajo, en el capítulo dos se muestran los resultados de un estudio de las diferentes áreas de proceso CMMI; con ello se determina el alcance de la integración; en el capítulo tres, se presentan cada una de las etapas de desarrollo con MDA, el modelo correspondiente, las herramientas usadas, la transformación al siguiente modelo, y los riesgos existentes en cada transformación; en el capítulo cuatro se detallan cada una de las actividades de desarrollo usando MDA, los recursos necesarios y los riesgos; en el capítulo cinco se presentan los artefactos propuestos en las

áreas de procesos con alto riesgo en un desarrollo MDD. En el capítulo seis, reconociendo algunas debilidades actuales que podrían afectar el proceso de desarrollo MDD, pero que no son propios de él, se analizan otros riesgos, y se plantean estrategias para mitigarlos. En el capítulo siete se describen algunos trabajos de otros autores que han vinculado CMMI con MDD. Por último, en el capítulo ocho se presentan los resultados de una encuesta realizada por la autora de la investigación, con el fin de validar la propuesta y la publicación de un artículo sobre la propuesta.

1.2. Motivación / Estado del Arte del Tema

1.2.1. Metodologías de desarrollo basadas en Modelos

El avance de las nuevas tecnologías y el aumento de la complejidad de los sistemas han traído nuevos problemas que resolver a la Ingeniería de Software. Esto ha conducido a los investigadores a pensar en nuevas metodologías, herramientas y lenguajes que permitan realizar los procesos de la Ingeniería de Software de forma eficiente. El uso de modelos en el desarrollo de software surge como respuesta a esta necesidad, buscando lograr al final la generación automática del código. MDD y MDE son dos de estos modelos.

- **Model-driven engineering - MDE o Ingeniería dirigida por modelos - IDM:** Basa el desarrollo de software más en modelos que en algoritmos e incluye actividades de desarrollo y de otros procesos adicionales de la ingeniería del software, como la evolución del sistema o el model-driven reverse engineering.

- **Model Driven software Development - MDD o Desarrollo de Software Dirigido por Modelos:** "Promete mejorar el proceso de construcción de software basándose en un proceso guiado por modelos y soportado por potentes herramientas".(4). Su inicio se dio cuando en Gran Bretaña en 1.994 un consorcio de compañías del Reino Unido quiso construir sobre Rapid Application Development (RAD) y desarrollo iterativo.
  El MDD está ganando cada vez más la atención de la industria y las comunidades de investigación. MDD induce a la automatización, la transformación de modelos y las técnicas de generación de código.

MDD : "Paradigma de desarrollo de software que utiliza modelos y cuyo objetivo es separar el diseño del sistema de la arquitectura de las tecnologías, para que puedan ser modificados independientemente". "Busca tres objetivos principales: Portabilidad, la Interoperabilidad y la Reutilización que eventualmente deberían conducir a un aumento en la productividad".
Los pilares básicos sobre los que se apoya MDD son los modelos, las transformaciones entre modelos y los lenguajes específicos de dominio.(24)

Beneficios del MDD: La generación automática de código con menos errores puede aumentar la productividad, y el uso de una arquitectura fácil de mantener, hace que sea menos difícil la implementación de los cambios que traen las nuevas tecnologías. Además, las nuevas funcionalidades que adopte el negocio sólo requerirían el desarrollo del modelo específico de las funciones, debido a que la información necesaria para generar los artefactos de implementación ya ha sido capturada en las transformaciones y puede ser re-usada.
'

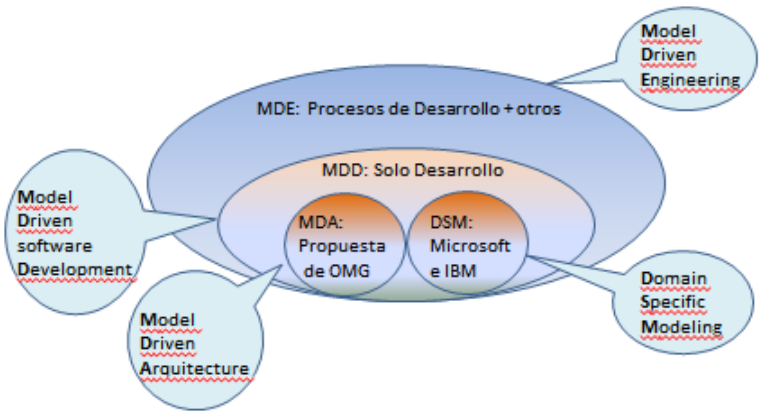


Figura 1.1: Relación entre diferentes modelos

.

Sus dos propuestas más conocidas son MDA y DSM.

MDA: Arquitectura Dirigida por Modelos (en Inglés: Model Driven Architecture) es la propuesta de MDD desarrollada por Object

Management Group - OMG. Tiende a enfocarse en lenguajes de modelado basados en estándares del OMG. (4). Tiene como principio fundamental separar la lógica de las aplicaciones de la plataforma software en la que dicha lógica vaya a ser implementada.

Definición de MDA acordada en Montreal en Agosto de 2002: MDA es una iniciativa que propone OMG para definir un conjunto de normas no propietarias que especifiquen tecnologías interoperables con los que cuenta el desarrollo dirigido por modelos con transformaciones automatizadas. MDA es parte de una tendencia más amplia dentro de la industria del software.

DSM: Propuesta de MDD utilizada por Microsoft, como parte del paquete de Visual Studio y también por IBM. Crea modelos para un dominio, utilizando un lenguaje focalizado y especializado para dicho dominio.

La figura 1.1 muestra la relación entre los diferentes modelos: MDE, MDD, MDA, y DSM. MDD define formalmente el modelo, mientras que MDA y DSM define las acciones que realizan los desarrolladores.

1.2.2. Modelos de Calidad

Sin lugar a dudas los procesos de calidad son reconocidos a nivel mundial porque bien llevados conducen al logro de los objetivos del negocio de manera más segura que sin ellos.

El SEI ha tomado la siguiente premisa de la gestión de procesos: "La calidad de un sistema o producto está muy influenciada por la calidad del proceso empleado para desarrollarlo y mantenerlo" y por esto ha definido CMMs que recogen esta premisa.(1).

Hacia el 15 de Septiembre de 2015, existían 5.014 certificaciones activas alrededor del mundo, repartidas entre 84 países. Esto representa un crecimiento de poco más del 24 % en tres años, en número de certificaciones desde el 2012 (4.031 certificaciones).(2)

El modelo actual de CMMI está escrito para los proyectos de todos los tamaños y formas, y para todas las disciplinas de ingeniería basadas en proyectos. Se ha aplicado con éxito en muchas empresas como lo demuestran las estadísticas tomadas de los resultados de evaluación presentados a la base de datos SAS del Instituto CMMI, ver figura 1.2 y 1.3. Esto, a pesar de los

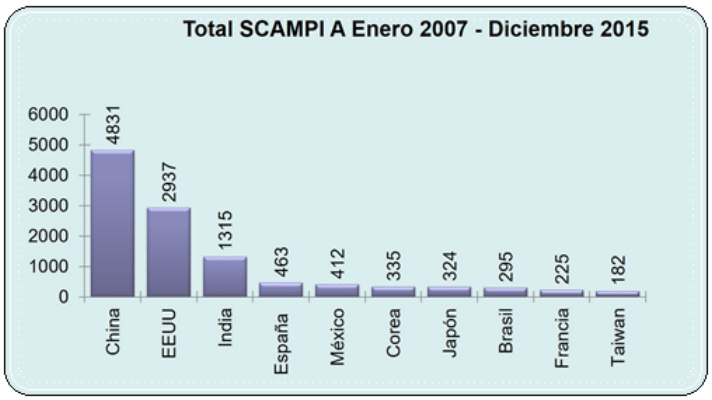


Figura 1.2: Número de Evaluaciones CMMI reportadas por país: Resultado de estudio, datos desde Enero del 2007 a diciembre del 2015.
(3).

cuestionamientos de algunos críticos acerca de varias dificultades como: Ser difícil de aplicar en la mediana y pequeña empresa, el modelo no puede responder a la pregunta de "cómo implementar las mejoras", y no fue concebido para aplicarse en empresas no comerciales. CMMI existe como una guía y no como un manual de instrucciones por lo tanto no responde a la pregunta "cómo implementar el modelo".

CMMI ha evolucionado y es muy amplio el ámbito en el que se puede aplicar para mejorar los procesos. Sin embargo, en éste trabajo nos enfocaremos en la constelación o CMMI-DEV: centrado en el desarrollo de productos.

Por todo lo anterior, se puede concluir que aplicar MDD y CMMI al proceso de desarrollo y mantenimiento de software, es una fórmula que asegura aún más la calidad de productos y servicios de software.

Siendo el modelo CMMI independiente de la metodología de desarrollo, este trabajo consistirá en traer las actividades de desarrollo definidas en CMMI-DEV, al desarrollo mediante MDD, más concretamente utilizando el enfoque Model Driven Architecture - MDA. Dejando para un posterior trabajo responder a la pregunta "Cómo validar el cumplimiento de estas actividades, en la evaluación del proceso de desarrollo".

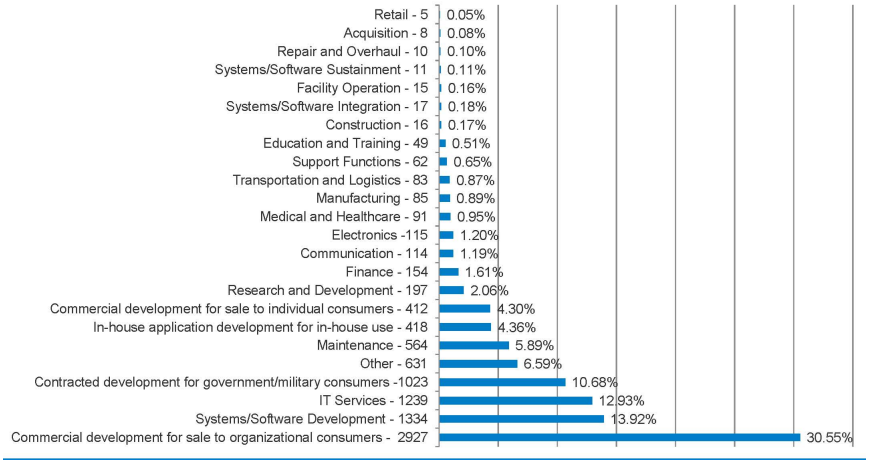


Figura 1.3: Caracterización de las empresas con CMMI: Resultado de estudio, datos desde Enero del 2007 a Marzo del 2014, para 9.580 evaluaciones CMMI-SCAMPI A

(3).

1.2.3. Resumen y conclusiones del capítulo

En éte capítulo se han definido conceptos relacionados con el modelo de calidad CMMI, el paradigma de desarrollo de software Dirigido por Modelos, MDD, y la propuesta de MDD desarrollada por Object Management Group - OMG: La Arquitectura Dirigida por Modelos (en Inglés: Model Driven Architecture) - MDA.

Los modelos MDD y CMMI, aportan de manera significativa a la mejora de los procesos de desarrollo de software desde dos aspectos diferentes: MDD utiliza los modelos, las transformaciones entre modelos y los lenguajes específicos de dominio, para lograr la generación automática del código, y aumentar la productividad, y CMMI busca producir productos y servicios de calidad, proveyendo los elementos necesarios para un proceso de desarrollo efectivo.

A partir del relevamiento bibliográfico realizado, se pudo comprobar la vigencia de los modelos CMMI y MDD y el respaldo que tienen los modelos en la comunidad de investigadores, docentes, desarrolladores, casas de software y consultores de TI.

Capítulo 2

Áreas de Proceso de CMMI a considerar

Elementos que contribuyen a la construcción del producto: El proceso, la tecnología, las personas.

2.1. Áreas de Procesos en CMMI

Los modelos CMMI reúnen las mejores prácticas de gestión de proyectos y de ingeniería que las organizaciones pueden utilizar para ayudarles a satisfacer y superar sus objetivos estratégicos y operativos. Su área de aplicación es muy amplia, pero para el tema que concierne a esta investigación nos concentraremos en las partes que están relacionadas con las actividades de desarrollo, reunidas en el modelo CMMI-DEV, que proporciona un conjunto completo e integrado de guías para desarrollar productos y servicios de calidad con el fin de cumplir las necesidades de clientes y usuarios finales. Existen otros dos modelos CMMI, constelaciones - o colecciones de componentes CMMI - que no atenderemos en esta investigación: El modelo CMMI-ACQ que se enfoca en las actividades para iniciar y gestionar la adquisición de productos y servicios y el modelo CMMI-SVC que atiende las actividades para proporcionar servicios de calidad al cliente y a los usuarios finales.

En este punto es importante aclarar que CMMI es un modelo; no una metodología. Por ello no tiene que responder a la pregunta "Cómo implementarlo?". CMMI sigue un enfoque de mejora de procesos.

> **Un modelo de procesos** es un conjunto estructurado de elementos que describen características de procesos efectivos y de calidad. Un modelo indica "Qué hacer", no "Cómo hacer", ni "Quién lo hace". Ej: CMMI-DEV.

La última versión de CMMI para el desarrollo, hasta la fecha - Febrero del 2017 -, es la versión CMM-DEV V1.3, que fue publicada en el 2010. En este modelo se definen cuatro niveles de madurez, cada uno con sus áreas de procesos PA. Las áreas de proceso que componen el modelo CMMI en su versión V1.3 se muestran en la tabla 2.1.

> **Un área de proceso** es un conjunto de prácticas relacionadas, que cuando son implementadas en conjunto, cumplen una serie de objetivos considerados importantes para hacer mejoras en esa área.

Una empresa puede abordar el modelo CMMI desde dos representaciones: La representación escalonada, que define para cada nivel de madurez un conjunto de áreas de proceso, y la representación continua que define el área a mejorar para alcanzar cierto nivel para cada una de las categorías de áreas de proceso.

Gráficamente la representación escalonada de CMMI se aprecia leyendo la tabla 2.1 de forma horizontal; es decir, se debe cumplir para cada nivel, con las áreas de proceso necesarias. Si se observa la tabla de manera vertical, se aprecia la representación continua, en ella se determina para cada categoría cuales son las áreas de proceso necesarias para lograr un determinado nivel.

CMMI-DEV contiene 22 áreas de proceso. De esas áreas de proceso, 16 son áreas de proceso base, 1 es un área de proceso compartida y 5 son áreas de proceso específicas de desarrollo.

Todas las prácticas del modelo CMMI-DEV se centran en las actividades de una empresa donde se desarrolla. Cinco áreas de proceso se centran en las prácticas específicas del desarrollo: Gestión de requerimientos (RD), Solución técnica (TS), Integración del producto (PI), Verificación (VER) y Validación (VAL).(1). Estas cinco áreas de proceso las analizaremos en este trabajo, por ser MDD un modelo de desarrollo. Pero también se analizarán las siguientes áreas: Gestión de la Configuración (CM), Seguimiento y Control de Proyecto (PMC), Planificación de Proyecto (PP), Aseguramiento de la Calidad de

Niv.				
5			Gestión del Rendimiento de la Organización (OPM)	Análisis Causal y Resolución (CAR)
4		**G**estión cuantitativa del Proyecto (QPM)	**R**endimiento de procesos de la Organización (OPP)	
3	**D**esarrollo de requerimientos (RD)..... **S**olución técnica (TS)..... **I**ntegración del producto (PI)..... **V**alidación (VAL) .... **V**erificación (VER)	**G**estión de Riesgos (RSKM)..... **G**estión integrada del proyecto (IPM)	**E**nfoque en procesos de la Organización (OPF) **D**efinición de procesos de la Organización (OPD) **F**ormación en la Organización (OT)	**A**nálisis de decisiones y Resolución (DAR)
2		**P**lanificación del proyecto... (PP) **M**onitorización y Control del Proyecto (PMC)..... **G**estión de Requerimientos (REQM) ....**G**estión de acuerdos con proveedores (SAM)		**G**estión de configuración (CM) **A**seguramiento de la calidad del proceso y del producto (PPQA) .... **M**edición y análisis (MA)
	Ingeniería	**Gestión de proyectos**	**Gestión de procesos**	**Soporte**

Tabla 2.1: Modelo CMMI por Nivel y Categoría. Estos procesos son los que por experiencia han demostrado ser efectivos.Fuente propia

Proceso y Producto (PPQA), y Gestión de Riesgos (RSKM), por tener una actividad con un manejo especial en un desarrollo dirigido por modelos.

A continuación se muestra un resumen de cada una de las áreas de proceso que se analizarán, extraídos de (1), y en el recuadro interno se indica cuál es el tema perteneciente a esa área que debería tratarse debido a una necesidad propia de un desarrollo MDD.

El área de procesos Gestión de Requerimientos (REQM) "Identifica las necesidades del cliente y traslada estas necesidades a los requerimientos del producto. Suministra estos requerimientos al área de proceso de Solución técnica, donde son convertidos en arquitectura, diseño y otros componentes del producto. Mantiene los requerimientos, describe las actividades para obtener y controlar sus cambios, y garantiza que los planes y datos relevantes se mantengan actualizados. Además, proporciona la trazabilidad de los requerimientos de los del cliente hasta los requerimientos del producto y de los componentes de producto".(1)

Para MDD ésta área debe garantizar la **trazabilidad bidireccional** de los requerimientos.

El área de procesos Solución Técnica (TS) "desarrolla los paquetes de datos técnicos relativos a los componentes de producto para que se utilicen en el área de proceso Integración del Producto o Gestión de Acuerdos con Proveedores. Aquí se examinan soluciones alternativas para seleccionar el diseño óptimo basado en criterios establecidos. El área de proceso Solución Técnica se basa en las prácticas específicas del área de proceso Verificación para realizar la verificación del diseño y las revisiones durante el diseño y antes de la construcción final".(1)

Para MDD ésta área debe atender **la reutilización de componentes** y determinar cuáles son las **herramientas MDD necesarias** para el desarrollo del proyecto.

El área de procesos Integración del Producto (PI), "contiene las prácticas específicas asociadas con la generación de una estrategia de integración, reuniendo los componentes de producto y entregando el producto al cliente. Aquí se verifican las interfaces para asegurar que cumplen con los requerimientos de interfaz suministrados por el área de proceso desarrollo de requerimientos. Utiliza prácticas de las áreas de verificación y Validación".(1)

Para MDD ésta área debe garantizar que **los artefactos y modelos utilizados cumplan los estándares y se mantenga la integridad del sistema.**

El área de proceso Verificación (VER), "asegura que los productos de trabajo seleccionados cumplan con los requerimientos".(1).
La verificación es generalmente una revisión de lo que se dice que se va a hacer frente a los requerimientos (en el papel).

Para MDD ésta área debe asegurar que para **cada una de las actividades de transformación de modelos se verifique si se están cumpliendo los requerimientos del negocio.**

El área de proceso Validación (VAL), "valida de manera incremental los productos frente a las necesidades del cliente."(1).
La validación trata de asegurar que lo que se construyó (o se planea construir) funcione cuando el usuario lo utilice. Si bien esto también se puede hacer "en el papel", es más común validar prototipos, y modelos.

Para MDD ésta área debe garantizar que **además de los requerimientos, los artefactos MDD construidos, también sean validados y testeados.**

El área de proceso de la Gestión de Configuración (CM) , "su propósito es establecer y mantener la integridad de los productos de trabajo utilizando la identificación de la configuración, el control de la configuración, el informe del estado de la configuración y las auditorías de la configuración."(1).

Para MDD ésta área debe **definir modelos de trazabilidad tales como controladores de la transformación de los modelos.**

El área de proceso Seguimiento y Control de Proyecto (PMC) , "El propósito de esta área es proporcionar una comprensión del progreso del proyecto para que se puedan tomar las acciones correctivas apropiadas, cuando el rendimiento del proyecto se desvíe significativamente del plan."(1).

Para MDD ésta área debe garantizar **la participación de todos los actores, en especial la del Arquitecto a lo largo de todo el proceso de desarrollo.**

El área de proceso Planificación del Proyecto (PP) , "El propósito de esta área es establecer y mantener planes que definan las actividades del proyecto. Sus metas son: Establecer las estimaciones, desarrollar un plan del Proyecto y obtener el compromiso del plan."(1).

Para MDD es importante que durante la actividad de **planificación de los recursos, se considere el conocimiento que el equipo de desarrollo debe tener acerca del Proceso de Desarrollo Guiado por Modelos.**

El área de proceso Aseguramiento de la Calidad del Proceso y del Producto (PPQA) , "El propósito de esta área es el de proveer al staff y gerencia de una visión y comprensión objetiva de los procesos y productos de trabajo asociados. Incluye las siguientes actividades: Evaluación objetiva de los productos y procesos, identificación y documentación de problemas, retroalimentación de resultados, y direccionamiento de problemas"(1).

Para MDD es importante seguir un Modelo de **trazado de los requerimientos para evitar que sean alterados dada la continua transformación de los Modelos.**

El área de proceso Desarrollo de Requerimientos (RD) , "tiene como propósito deducir, analizar y establecer los requisitos de cliente, de producto y de componentes de producto."(1).

En MDD la transformación de modelos puede iniciar antes o después de afinar los requerimientos. En caso de iniciar desde el modelo CIM, que contiene las reglas del negocio, es deseable el uso de herramientas que permitrán la transformación automática a PIM.

El área de proceso Gestión de Riesgos (RSKM) , "El propósito de esta área es identificar problemas potenciales antes de que ocurran, para que las actividades de tratamiento de riesgos puedan planificarse e invocarse según sea necesario a lo largo de la vida del producto o del proyecto para mitigar los impactos adversos sobre la consecución de objetivos."(1).

Para MDD es importante **gestionar los riesgos luego de cada una de las transformaciones de los modelos.**

Lo que sigue ahora, es identificar los riesgos para cada una de estas áreas de proceso al utilizar metodologías de desarrollo de modelos y determinar cómo se deben gestionar estos riesgos para alcanzar los objetivos de calidad, y de rendimiento en el desarrollo.

2.1.1. Resumen y conclusiones del capítulo

En éste capítulo, teniendo en cuenta que éste trabajo debe plantear los artefactos necesarios para que un proyecto MDD logre soportar totalmente las prácticas específicas de un conjunto de áreas de proceso, niveles 2 y 3, de CMMI DEV 1.3, se ha hecho una revisión de cada una de las áreas de procesos CMMI relacionadas con las actividades de desarrollo de software, buscando identificar las mejores prácticas CMMI-DEV, que se deben cumplir en el desarrollo de software bajo el paradigma MDD.

Las siguientes prácticas que se analizarán en el resto de la investigación:
Nivel 2

- CM Gestión de la Configuración
- PMC Seguimiento y Control de Proyecto
- PP Planificación de Proyecto
- PPQA Aseguramiento de la Calidad de Proceso y Producto
- REQM Gestión de Requerimientos

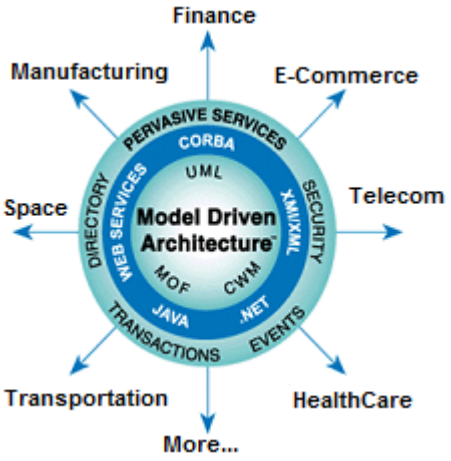
Nivel 3

- PI Integración de Producto
- RD Desarrollo de Requerimientos
- RSKM Gestión de Riesgos
- TS Solución Técnica
- VER Verificación.

Además, se determinó el tema perteneciente a esa área que debería tratarse debido a una necesidad propia de un desarrollo MDD, para analizar en el siguiente capítulo el riesgo existente.

Capítulo 3

Proceso de desarrollo MDA



El modelado de sistemas software es una técnica para tratar con la complejidad inherente a estos sistemas.

MDA es un framework cuyo objetivo es obtener beneficios que mejoren la portabilidad, la interoperabilidad, la reutilización y la productividad en el desarrollo de software, mediante la separación del diseño de la arquitectura de las tecnologías de construcción, y el uso de modelos que se transformen los modelos, hasta la obtención el código.

Los principios en los que se fundamenta MDA son la abstracción, la automatización y la estandarización. MDA plantea el siguiente proceso de desarrollo: Se parte de un modelo abstracto, independiente de la computación, o CIM, seguido de un modelo independiente de la plataforma (PIM), luego este modelo es transformado con la ayuda de herramientas en uno o más modelos específicos de la plataforma (PSM), y finalmente cada PSM es transformado

en código. De esta forma, MDA utiliza las transformaciones entre modelos (PIM a PSM, PSM a código), y se vale de herramientas para automatizar estas tareas. Estas herramientas de transformación son, de hecho, uno de los elementos básicos de MDA.

La figura 3.1 muestra la secuencia de los procesos en el desarrollo utilizando el framework MDA, y el rol del principal actor en cada uno de los procesos. En el capítulo 4 se detallarán los demás roles y las funciones de cada uno.

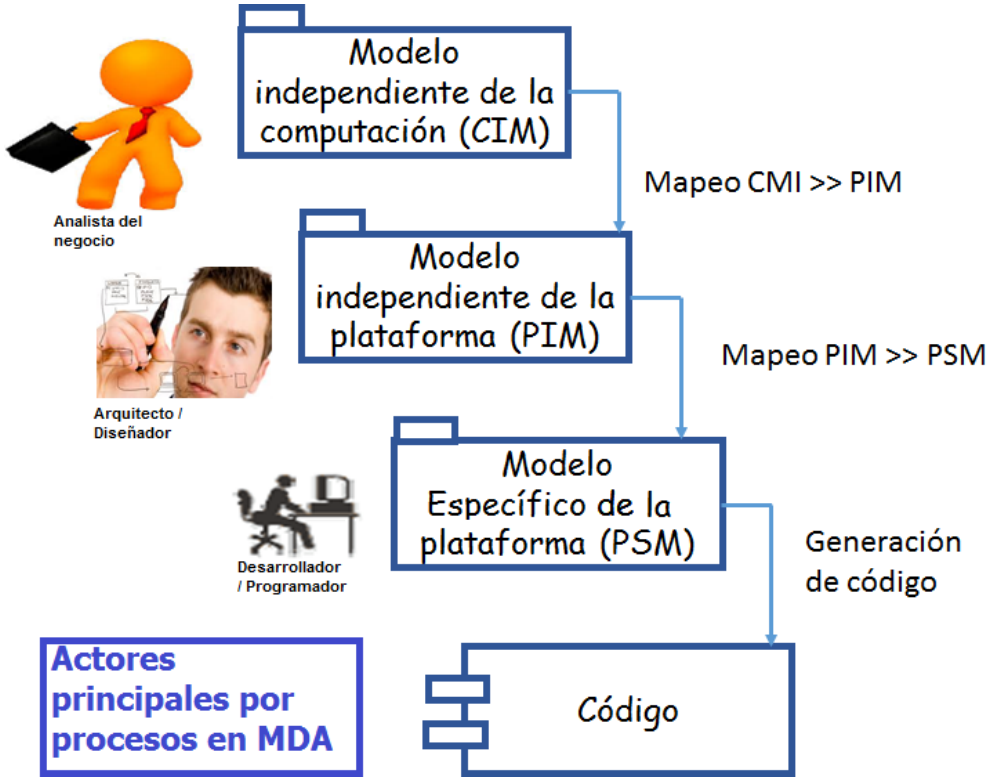


Figura 3.1: Procesos y roles de los actores principales en cada proceso en MDA

.

A continuación se presentará para cada una de las etapas de desarrollo con MDA, el modelo correspondiente, las herramientas usadas, la transformación al siguiente modelo, y los riesgos existentes en cada transformación.

Muchas de las herramientas existentes se especializan en alguna o algunas de las actividades de desarrollo por modelos, pero raramente soportan el ciclo completo MDD; por ello su mención se hará de acuerdo a cada uno de los procesos.

3.1. Construcción del modelo independiente del cálculo o de la computación, CIM

Modelo independiente de la computación o Modelo del Dominio, CIM: Es el modelo más abstracto de todos los usados en MDA, y describe la lógica del dominio del negocio desde una perspectiva independiente de la computación. El CIM juega un papel importante como puente entre los que son unos expertos en el dominio del problema, y sus requerimientos y aquellos que son expertos en el diseño y construcción de artefactos software.

Como todo modelo, un CIM debería cumplir con las siguientes características (22):

- **Abstracto**: Eliminar u ocultar el detalle que es irrelevante para un punto de vista de los patrocinadores, usuarios y clientes del sistema. Esto permite comprender la esencia más fácilmente.
- **Comprensible**: Reducir la cantidad de esfuerzo intelectual necesario para la comprensión.
- **Exacto**: Ser una verdadera representación del sistema que se modela.
- **De bajo costo**: Ser más barato de construir y analizar que el sistema modelado.

3.1.1. Herramientas para la construcción del modelo CIM

El uso de herramientas para la modelación, ayuda a las organizaciones a optimizar el desempeño de sus negocios mediante la revisión, documentación, automatización y mejora continua de los procesos de negocio, incrementando la eficiencia y reduciendo los costos.

Los siguientes son estándares internacionales para el modelado de procesos:

- **OMG (Object Management Group)**

  SPEM (Software Process Engineering Metamodel): Su especificación fue publicada en el 2002 por el consorcio OMG. Proporciona una serie de elementos para representar de manera estandarizada los métodos, ciclos de vida, roles, actividades, tareas y productos de trabajo utilizados en Ingeniería de software.

  "El diseño de procesos con SPEM no es una tarea sencilla debido a

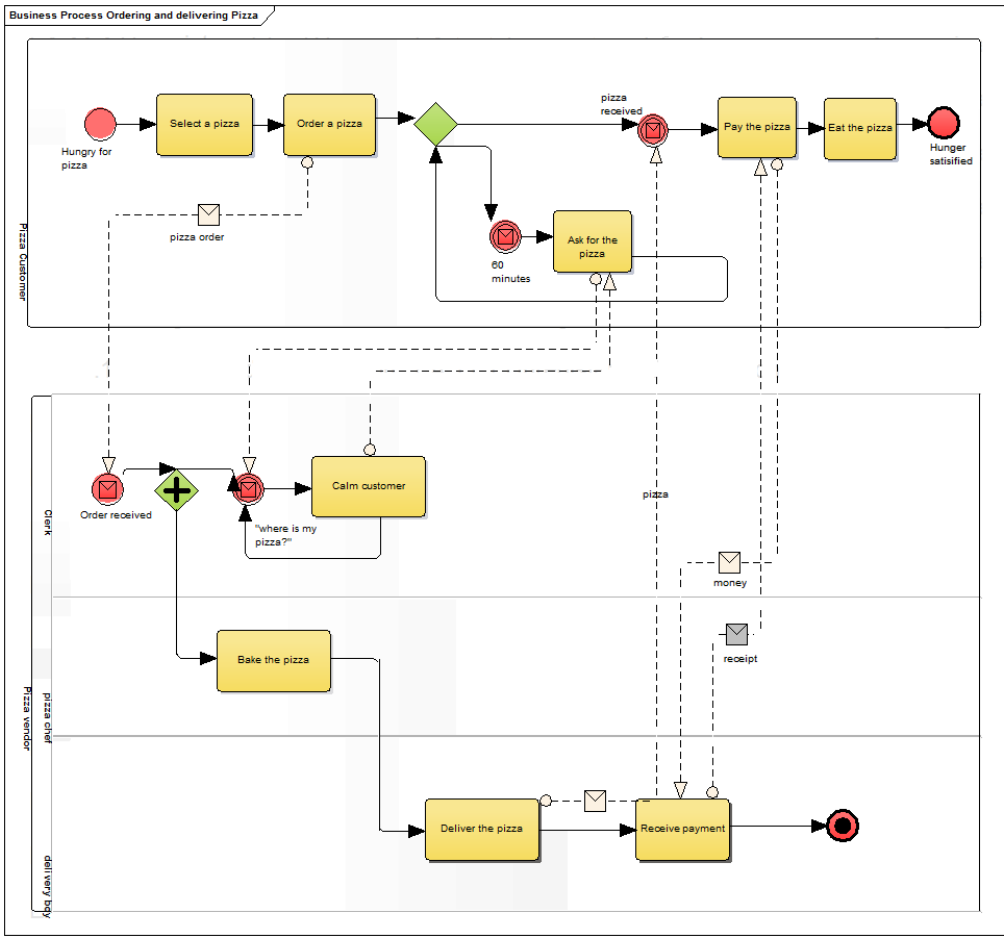


Figura 3.2: Ejemplo de modelo BPMN. Orden y liberación de una pizza. Fuente Ejemplo BPMN incluido en el paquete Enterprise Architect

.

la complejidad del propio lenguaje, que requiere un profundo entendimiento de la especificación. Además, el lenguaje no ha conseguido suficiente nivel de aceptación en la industria y hasta la fecha, se utiliza casi solamente en ámbitos académicos y de investigación."(12).

Diagramas de Actividad de UML (Unified Modeling Language): De Object Management Group (OMG), es un lenguaje de modelado de propósito general aunque en sus raíces están el modelado de sistemas de software y, específicamente, programación orientada a objetos. Puede ser usado para elaborar el modelo de negocios.

- **EPC (Event-driven Process Chain) o Cadena de procesos impulsada por eventos**: Es una técnica de modelado especialmente útil

cuando se trata de implementar un ERP. Desarrollado por August-Wilhelm Scheer en el Instituto de Ciencias de la Computación en la Universidad de Saarland a principios de 1990.

- **DFD (Data Flow Diagram)** : Diagramas de flujo que pueden tener dos estilos de estándares: el formato Yourdon/DeMarco (tipificado por círculos y flechas de flujo arqueadas) y el modelo Gane Sarson (tipificado por cuadrados redondeados y flechas de flujo más rectas).

BPMN (Business Process Modeling Notation)

El lenguaje de modelación Business Process Management Notation o BPMN, es un lenguaje notacional gráfico desarrollado por la BPMI (Business Process Management Initiative) y soportado por la OMG, para la modelización de procesos de negocio que permite representar el funcionamiento de la organización, en forma clara y precisa a través de la creación de un CIM y facilita la transformación del CMI en PIM usando el lenguaje de transformaciones estándar QVT (Query/View/Transformation).

Una ventaja de representar el modelo utilizando BPMN en lugar de UML, es que éste es más accesible a todo tipo de usuarios a la hora de su análisis. "La utilización del modelado de los procesos de negocio a través de BPMN, permite al analista mejorar la obtención y refinación de requerimientos, con elementos conocidos por los usuarios, no sólo por los expertos"(7).
La figura 3.2 muestra un ejemplo de modelo BPMN.

En cuanto a software para modelar procesos de negocio utilizando BPMN, existen varios productos en el mercado como:

- **Enterprise Architect de Sparx Systems**: Permite la generación de modelo BPMN a partir de diagramas UML.
- **Bizagi Process Modeler**: herramienta freeware de gestión de procesos ágil y fácil de utilizar que permite diseñar, diagramar, documentar y publicar los procesos utilizando el estándar BPMN.
- **ADONIS Community Edition** es una versión libre de la herramienta líder en BPM ADONIS. Está dirigido tanto a usuarios, que necesitan una herramienta intuitiva para la documentación de los procesos del negocio.
- **objectiF**: Desarrollado por microTOOL, basado en el Lenguaje de Modelado Unificado (UML) y el Business Process Modeling Notation (BPMN) permite modelar los procesos del negocio. La

generación de código se puede hacer para varios lenguajes de programación: Java, C , Visual Basic.NET y C ++. Objectif se puede utilizar en el entorno de desarrollo de Microsoft Visual Studio y Eclipse integración, permitiendo así a la ingeniería de ida y vuelta.

- **Together**: Borland Together principalmente soporta modelados en UML y la generación de código para Java, C, VB. NET, C, COBRA/IDL. El modelado de Procesos del Negocio se hace utilizando Notación de Modelado de Procesos del Negocio (BPMN, por siglas en ingles) y el Modelado de datos mediante UML 2.0 o diagramas Entidad-Relación.
- **Rational Rose XDE**: De IBM da soporte a la arquitectura de software o negocio y al modelado de datos.
- **Eclipse Modeling Framework (EMF)**: Es una plataforma abierta de propósito general para el desarrollo de software. La plataforma está constituida por una serie de proyectos básicos al que se añaden plugins que completan su funcionalidad. Las herramientas de Eclipse presentan cierta dificultad en su aprendizaje, pero se puede realizar una buena integración entre ellas, permitiendo crear nuevas herramientas de transformación de modelos de alto nivel de abstracción, independientes de la plataforma tecnológica. Bonita Open Solution es una herramienta disponible desde la plataforma Eclipse que permite al usuario modificar gráficamente los procesos de negocio siguiendo el estándar BPMN.

3.1.2. Transformación CIM en PIM

La transformación es el proceso que basado en una serie de reglas, define los mecanismos para el paso de un modelo origen a un modelo destino.

Luego de investigar sobre la transformación automática del CIM en PIM, se puede decir, que en la actualidad, son pocos los trabajos documentados que se encuentran sobre transformaciones CIM a PIM de manera automática, a pesar de la gran cantidad recursos, tiempo y esfuerzos dedicados a la investigación de MDA. "Como los modelos conceptuales y modelos de ingeniería apuntan a temas de distintos dominios, los modeladores raramente intentan relacionar uno con otro".(20).

"Lastimosamente, el proceso de desarrollo usando MDA, no considera la posibilidad de transformar el CIM en un PIM. Y aunque el CIM es bastante abstracto, parece necesario establecer un procedimiento que permita construir el PIM a partir del CIM, con la ayuda de una herramienta MDSD y un conjunto de reglas de transformación".(10).

A continuación se presentan las reglas a tener en cuenta en la transformación de un CMI a un PMI, según Y. Rhazali, Y. Hadi, y A. Mouloudi, (14), que podrían implementarse en el desarrollo de una herramienta de transformación usando QVT:

- De CMI a un diagrama de casos de uso:
  - Cada tarea del diagrama modelo de proceso que corresponde a una funcionalidad del sistema se transforma en caso de uso.
  - El colaborador, quien se da cuenta de los sub-procesos en el modelo de colaboración BPMN, se convierte en un actor del caso de uso que corresponde a las tareas de este sub-proceso.
  - Si hay "una puerta de enlace exclusiva" entre dos tareas, los casos de uso correspondientes se conectan por una relación "extend".
  - Si sólo hay un flujo de secuencia entre dos tareas, los casos de uso correspondientes se conectan por una relación "incluir".
  - No transformar el flujo de secuencia de retorno.
  - Cada sub-proceso del modelo de diagrama de colaboración es transformado a un paquete que incluye los casos de uso correspondientes a las tareas de este sub-proceso.
- Reglas de transformación de un modelo de procesos a un modelo de clases:
  - Transformar los de objetos de datos del diagrama de proceso a clases.
  - Si hay varios objetos de datos que tienen el mismo nombre, éstos deben ser transformados a una clase.

3.1.3. Riesgos en la construcción de un CIM y su transformación a PIM

"Cuando se cometen errores en las primeras etapas del desarrollo, su solución es más costosa. Por ello se busca que el modelo de negocios que se prepare sea rico y contenga la información necesaria para su transformación en PIM. El CIM debe ser construido sin ambigüedades y con mucha precisión. Por esto se recomienda utilizar técnicas y tecnologías en su preparación".(8).

Los siguientes son algunos de los riesgos en los que se podría incurrir al elaborar un CIM:

- Que el modelo CIM tenga bajo nivel de exactitud.
- Que no contenga todas las reglas del negocio requeridas para el modelo del proceso de negocio que se modela.

- Que no se incluyan completamente todos los actores en el modelo: primarios o tomadores de decisiones, secundarios, o que participan en la toma de decisiones, y terciarios, o que producto del proceso de toma de decisiones, llevan a cabo las actividades o subprocesos operacionales.(9).

Las siguientes son algunas reglas a tener en cuenta en la construcción de un buen CIM, según Y. Rhazali, Y. Hadi, y A. Mouloudi usando modelos de colaboración y de procesos BPMN, dentro de su propuesta de cómo automatizar en CMI en PMI desde un enfoque generalizado:

- Modelos BPMN de colaboración:
  - "Definir los principales procesos, y no los sub-procesos complejos. Se recomienda que los sub-procesos estén compuestos de alrededor de 4 a 12 tareas.
  - Si un sub-proceso consiste en menos de 4 tareas, o representa una operación complementaria a otro sub-proceso, fusionar dos sub-procesos en uno solo.
  - Evitar al máximo posible, la representación de las tareas y presentar las tareas sólo manuales.
  - El modelo no describe todos los casos posibles y caminos, sino que simplemente presenta una descripción de la secuencia de actividades de los procesos de negocio más comunes.
  - Centrarse en sub-procesos y sus secuencias.
  - Identificar el máximo posible de los actores que interactúan y que colaboran en la consecución de los procesos del negocio.
  - Evitar al máximo posible, la representación de las puertas de enlace en este nivel.
- Modelos de procesos BPMN:
  - Detallar individualmente cada sub-proceso en un modelo con sus diversas tareas.
  - No representar las tareas manuales del modelo de colaboración.
  - Representar las puertas de enlace en este modelo.
  - Enriquecer este modelo con otras representaciones.
  - Añadir un objeto de datos en la salida de cada tarea." (14).

3.2. Construcción de un Modelo Independiente de la Plataforma (Platform Independent Model o PIM):

El PIM (Modelo Independiente de la Plataforma) de MDA, es el segundo modelo en jerarquía de MDA, que describe todos los aspectos funcionales de la aplicación independientemente de la plataforma software en la que vaya a implementarse y ejecutarse.

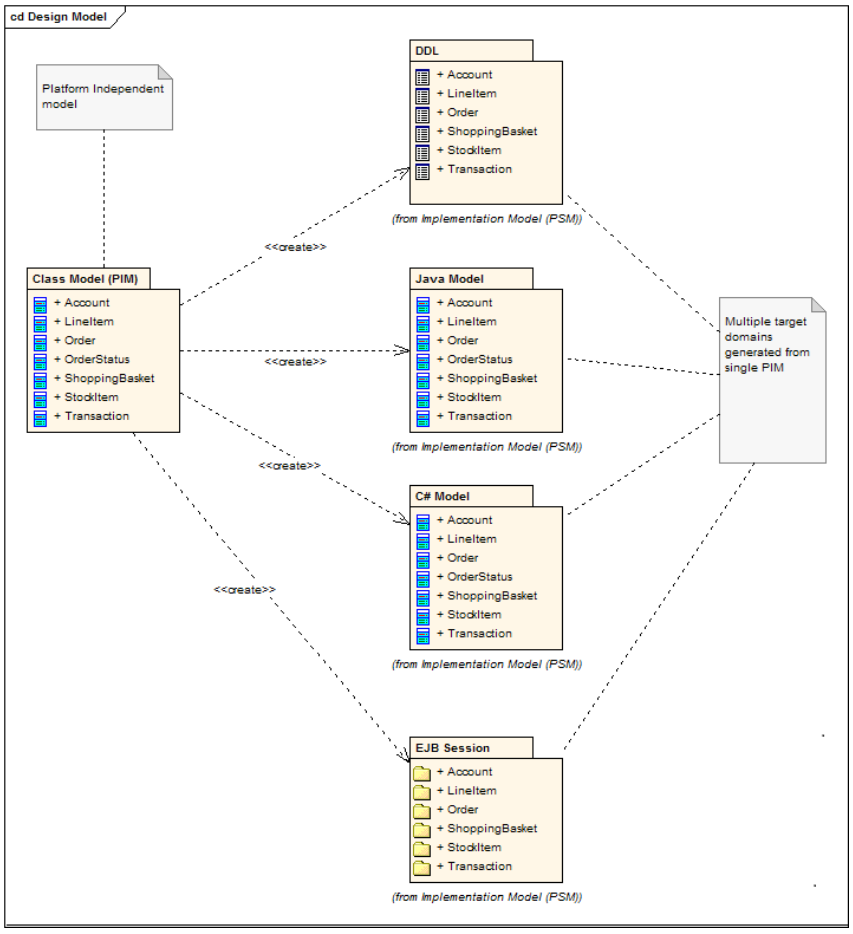


Figura 3.3: Un único PIM transformado en diferentes PSM

El modelado independiente de la plataforma exige una representación suficientemente general como para capturar la semántica de muchos ámbitos

diferentes, pero lo suficientemente precisa para apoyar la eventual transformación en código.(11).

Al no incluir detalles específicos de una tecnología determinada, éste modelo es útil en dos aspectos:

- Es fácilmente comprensible para los usuarios del sistema, y por lo tanto, les resulta más sencillo validar la corrección del sistema.
- Facilita la creación de diferentes implementaciones del sistema en diferentes plataformas, dejando intacta su estructura y su funcionalidad básica. Ver figura 3.3.

3.2.1. Herramientas para la construcción del modelo PIM

Los modelos PMI son modelos de clases que se representan más comunmente como modelos de clase UML. Las siguientes son herramientas que permiten modelar PIM:

- **OLIVANOVA Modeler**: Permite crear, editar y validar sus PIMs, convierte PIM en PSM de manera automática, y éstos a código. Fue desarrollado por la empresa CARE Technologies (Computer Aided Requirements Engineering Technologies, S.A), junto con el apoyo del grupo de investigación OO-Method. Su software es comercial.
- **Architect**: Satisface todos los requisitos básicos para ser considerado compatible con MDA, y proporciona muchas características adicionales para apoyar este nuevo enfoque de desarrollo. Permite la transformación modelo a modelo PMI a PSM y mapear hacia esquemas C , DDL, EJB, Java, JUnit, NUnit, WSDL, XSD y XML.
  Enterprise Architect está disponible en cuatro ediciones: Corporate Floating, Corporate, Professional y Desktop.
- **ArcStyler de Interactive Objects - Software**: En ArcStyler el PIM pasa directamente a código sin que exista un PSM que pueda cambiar el desarrollador, sin embargo tiene un punto a favor, es una herramienta que permite generar código para diferentes plataformas mediante la arquitectura CARAT (Cartuchos MDA). En ArcStyler el esfuerzo de desarrollo es mayor, ya que el programador utiliza PIMs marcados en reemplazo de PSMs así como también la creación de etapas intermedias para poder llegar al modelo de destino (código o PSM). (13)
- **AndroMDA**: Es software Open Source. Aunque se basa en MDA, no basta sólo con los modelos para llegar a un despliegue, es necesario que el desarrollador intervenga el código y por lo tanto requiere que éste tenga un buen conocimiento de la plataforma.

- **OptimalJ de Compuware**: Fue uno de los primeros entornos de desarrollo avanzada para implementar MDA desde el 2001, pero Compuware decidió en 2008 dejar de OptimalJ.
- **Graphical Modeling Framework (GMF)**: Es software Open Source. Posibilita la creación de una aplicación de modelado para BPMN o UML2, que después pueden ser usados para crear los modelos reales en otros lenguajes.

3.2.2. Transformación PIM en PSM

La transformación de PIM a PSM puede ser hacerse de las siguientes formas:

- Manualmente transformando el PIM en PSM.
- Generando de manera automática un esquema general del PSM y completándolo manualmente.
- Totalmente de manera automática.

"MDA al permitir la doble transformación automática PIM-PSM y PSM-código fuente hace que el esfuerzo se centre en los modelos de alto nivel y al regenerar el resto de los modelos, la trazabilidad está garantizada. Esto hace que las iteraciones dentro del ciclo de vida sean más eficaces para afrontar estos cambios, minimizando de esta manera los riegos asociados" (15)

Al transformar el PIM en PSM, se debe tener en cuenta la información no funcional, que afecta la elección de la plataforma donde se implementará. Por ejemplo, el número de personas que concurrentemente utilizará el sistema.

3.2.3. Riesgos en la construcción de un PIM y su transformación a PSM

Los riesgos a nivel operativo que se centran en los equipos de desarrollo en un desarrollo tradicional, se mitigan en MDA de manera sustancial, pero aún persisten los riesgos asociados a una mala definición de la arquitectura, al bajo nivel de exactitud del modelo trazado, o las decisiones sobre los modelos destino generados por la transformación. (18).

Las transformaciones para pasar de un modelo a otro, se constituyen en los pilares de MDA. Por ello, las herramientas de transformación de modelos juegan un papel importante en el desarrollo, y vale la pena revisar las características mínimas que se espera deben cumplir las herramientas de transformación de modelos.

Es recomendable que las herramientas de construcción y transformación de modelos cumplan las siguientes características:

- Que permitan a los desarrolladores ajustar las transformaciones.
- Que permitan conocer el origen de cualquier elemento del modelo: Trazabilidad.
- Que los cambios que se hagan en el modelo inicial se conserven cuando sea necesario volver a generar el modelo: Consistencia incremental.

El PIM debe ser conductualmente completo, definiendo la lógica de negocio en términos de acciones abstractas. (4)

3.3. Construcción de un Modelo específico de la Plataforma (Platform-specific models o PSM):

"Es posible generar un código a partir de un PIM, pero la mayoría de los proyectos de desarrollo consideran varios aspectos del sistema antes de sumergirse en la codificación. Estas decisiones se refieren a la elección de la tecnología de despliegue, incluyendo el sistema operativo, la programación lenguaje y de alto nivel protocolos. MDA anticipa la necesidad de tomar estas decisiones y los separa de la generación de código mediante la introducción del PSM como capa intermedia". (11).

El PSM (Modelo Específico de la Plataforma) de MDA, es un modelo del sistema con detalles específicos de la plataforma en la que será implementado. Se genera a partir del PIM, así que representa el mismo sistema pero a distinto nivel de abstracción.

3.3.1. Herramientas para la construcción del modelo PSM

OLIVANOVA Model Execution define modelos de aplicación, equivalentes al concepto de PSM definido en MDA, para las diferentes capas lógicas de una aplicación. Para cada modelo de aplicación definido un OLIVANOVA Transformation Engine automatiza la transformación de modelo conceptual a modelo de aplicación (PIM-PSM) y del modelo de aplicación al código (PSM-PIM).

Enterprise Architect también permite realizar transformaciones de PMI a PSM y además, crea enlaces internos (Dependencias de transformación) entre cada PSM creado y el PIM inicial. Esto es esencial, porque permite sincronizar el PIM y el PSM muchas veces; por ejemplo, se puede añadir un

nuevo atributo a una clase PIM y sincronizar para que se cree nueva columna en el modelo de datos.

ArcStyler utiliza PIMs "marcados" en reemplazo de PSMs.

La figura 3.3 muestra diferentes PSM generados a partir del mismo PIM.

Cabe anotar que si se necesitan transformaciones distintas a las herramientas disponibles, se deberá hacer uso de herramientas de creación y modificación de transformaciones. Es en este punto donde toma importancia un lenguaje de definición de transformaciones estándar.

3.3.2. Transformación PSM en código

Debido a que un PSM, es demasiado abstracto para la compilación en un idioma determinado, un entorno MDA requiere otro conjunto de reglas de transformación y una herramienta de "Desarrollo de software guiado por modelos", o MDSD, para generar el código a partir del PSM. Además, para que las transformaciones sean exitosas, se deben construir modelos de buena calidad y mantener el enfoque de modelado a lo largo de la vida del proyecto en lugar de ajustar el código.

Tal vez sea necesario que se generen varios PSM e ir refinándolos hasta obtener el que se transformará en código, y al final añadir las funcionalidades que no se puedan implementar en el PIM o en los PSM. También puede ser necesario que se construyan varios PSM para los diferentes módulos del sistema, en cuyo caso también sería necesario utilizar "Puentes", o herramientas de transformación, para que se comuniquen los diferentes PSM.

"Un esquema típico de desarrollo con MDA usando varios PSMs podría incluir PSM´s para:

- Un modelo relacional del sistema: que describa el esquema de base de datos del sistema mediante un diagrama de Entidad-Relación.
- Un modelo EJB, mostrando los aspectos relativos a la plataforma EJB.
- Un modelo Web para describir las interfaces Web del sistema".Ver figura 33.4 (17).

3.3.3. Riesgos en la construcción de un PSM y su transformación a código

Los riesgos en esta etapa están inmersos en las siguientes actividades:

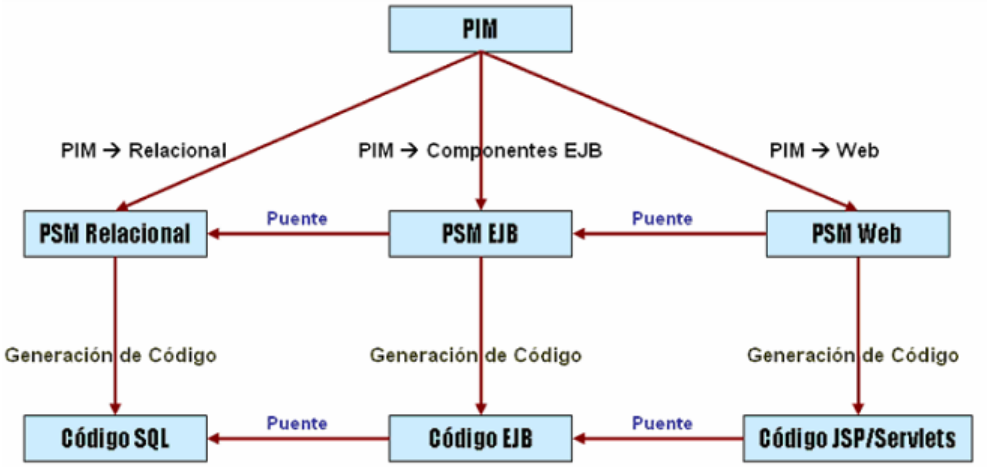


Figura 3.4: Comunicación de diferentes PSM en un sistema
. (17)

- Seleccionar un patrón de diseño,
- Ejecutar reglas de transformación,
- Verificar consistencia y completitud del modelo,
- Verificar escenarios de prueba.

Un riesgo que actualmente presentan las herramientas basadas en MDA es que no son fáciles de configurar inicialmente, ni es fácil definir las transformaciones para una plataforma. Este esfuerzo es compensado cuando se utilizan estas transformaciones para múltiples aplicaciones. En la figura 3.5 y figura 3.6 se muestran un comparativo de las herramientas que apoyan el desarrollo dirigido por modelos en alguna de sus fases.

3.3.4. Resumen y conclusiones del capítulo

En éste capítulo se hizo una descripción de cada una de las etapas de un desarrollo MDD: CIM, PIM, PSM y generación a código, y se nombraron algunas herramientas utilizadas en la transformación de cada modelo, al siguiente. Además, como mecanismo para determinar los artefactos para que un proyecto MDD logre soportar totalmente las prácticas específcas de un conjunto de áreas de proceso, niveles 2 y 3,de CMMI DEV 1.3, se indagó sobre los riesgos en la construcción y transformaciones de cada uno de los modelos.

En la búsqueda de herramientas, se detectaron falencias en las actuales

herramientas, como la falta de herramientas que transformen de manera automática el modelo CIM a PIM, y el alto grado de complejidad de las transformaciones con las herramientas actuales.

- AADL from Carnegie-Mellon Software Engineering Institute
- Acceleo an open source code generator from Obeo
- Actifsource
- Apollo for Eclipse from Gentleware
- ATLAS Transformation Language or ATL, a model transformation language from Obeo
- AndroMDA an open source MDA tool [1]
- anycode a free MDA plugin for Astah UML Community
- ArcStyler from Interactive Objects Software GmbH
- Artisan Studio from Atego
- ASCET from ETAS
- AtomWeaver from Isomeris
- CoCoViLa from Tallinn University of Technology
- CodeFluent Entities from SoftFluent
- DB-MAIN free framework from REVER
- ECO (Domain Driven Design) EnterpriseCoreObjects by CapableObjects.com
- Eclipse Modeling Framework (EMF)
- Enterprise Architect from Sparx Systems
- ER/Studio from Embarcadero Technologies
- Epsilon from the University of York
- Fujaba
- GenerateXY from DotXY
- Generic Eclipse Modeling System (GEMS)
- GeneXus a Knowledge-based, declarative, multi-platform, multi-language development solution
- Graphical Modeling Framework (GMF)
- HERMES from RWTH Aachen University
- HyperSenses and ANGIE from DELTA Software Technology
- Innovator from MID GmbH
- W4 EXPRESS (former. LEONARDI),XML Frames Some Very Rapid Applications Development Engines with same analysis files.
- MagicDraw from No Magic Inc
- ManyDesigns Portofino
- MERODE JMermaid from KU Leuven (educational)
- MetaEdit+ from MetaCase
- Mia-Studio from Mia-Software
- objectiF from microTOOL
- openArchitectureWare
- Open ModelSphere
- OptimaJ from Compuware
- PREEvision from Vector Informatik
- Real Time Developer Studio from PragmaDev
- Rhapsody from IBM
- RISE Editor from RISE to Bloome Software
- SCADE Suite from Esterel Technologies
- Sculpture Platform from Modelingsoft
- Select Architect from Select Business Solutions
- Simulink from MathWorks, see also Stateflow and Real-Time Workshop Embedded Coder, TargetLink
- Sirius an Eclipse Open Source project to create custom graphical modeling workbenches, from Obeo and Thales
- TASTE from European Space Agency combining several modeling technologies
- Together Architect from Borland
- PolarSys (Open Source-Tool)
- Umple from the University of Ottawa
- Uniface from Compuware
- Visual Paradigm solutions from Visual Paradigm
- XComponent, a user-friendly solution
- YAKINDU Statechart Tools open source tool build on top of Eclipse

Figura 3.5: Herramientas para MDE. Fuente "Model-driven engineering", Wikipedia.

Herramienta / Característica	Comercial	Código abierto	CIM	PIM	PSM	CODIGO	
Adonis		X	X				http://www.adonis-community.com/
AndroMDa		X			X	X	http://www.andromda.org/whatisit.html
Bizagi		X	X				http://www.bizagi.com/es/bpm-suite-es/productos/modeler
Bonita BPM		X					http://www.bonitasoft.com/
Together, de Borland	X		X		X	X	http://www.borland.com/Products/Requirements-Management/Together/
Eclipse Modeling Framework (EMF), BPMN2 Modeler,		X	X	X	X	X	http://www.eclipse.org/modeling/emf/
ObjectiF	X			X	X	X	http://www.microtool.de/en/objectif-model-driven-development/
Enterprise Architect, de Sparx Systems	X		X	X	X	X	http://www.sparxsystems.com/products/ea/downloads.html
Rational Rose XDE, de IBM.	X		X	X	X	X	http://www-03.ibm.com/software/products/es/rosemod
ArcStyler	X			X		X	www.arcstyler.com

Figura 3.6: Herramientas para MDD. Fuente propia.

Capítulo 4

Actividades de desarrollo en MDA, Personas a cargo y Controles

"Zapatero a tus zapatos".
Refrán popular.

En éste capítulo se detallarán cada una de las actividades de desarrollo usando MDA, los recursos y los controles que se podrían aplicar en cada momento teniendo en cuenta los riesgos analizados en el capítulo 3.

Se seguirá un orden cronológico de las actividades involucradas en el desarrollo MDA:

- Generación de los Modelos Independientes de la Computación - CIM.
- Construcción de un Modelo lndependiente de la Plataforma (Platform lndependent Model o PlM).
- Construcción de un Modelo dependiente de la Plataforma (Platform-specific model o PSM).

- Transformación PSM en código.

Para ilustrar cada área de proceso, se utilizará un gráfico que reúne los actores involucrados, el nombre de la actividad central y las salidas de cada actividad del proceso. ver figuras 4.2 a 4.5.

Para los representar los actores de cada proceso se utilizará la siguiente convención:

Figura 4.1: Convención para el recurso humano necesario en cada grupo de actividades. Fuente. Propia.

4.1. Generación de los Modelos Independientes de la Computación - CIM)

En el nivel de requerimientos dentro de un proyecto de desarrollo con MDA, se realizan todas las actividades que permiten obtener los modelos fuente de una transformación, cuando las actividades de modelado inician en éste nivel.

La gráfica 4.2 muestra el proceso de Generación de los Modelos Independientes de la Computación, o CMI´s. Este se corresponde con los niveles de afinación de requerimientos y construcción del modelo de requerimientos. Los objetivos de cada grupo de actividades son:

- **Identificar elementos del contexto del problema**: "Esta actividad está compuesta por acciones que siguiendo un flujo determinado, guían al equipo de desarrollo conformado por el patrocinador, los stakeholders, y el arquitecto, en la planeación inicial del proyecto y la identificación de productos de trabajo que formalizan los CIM". (18).

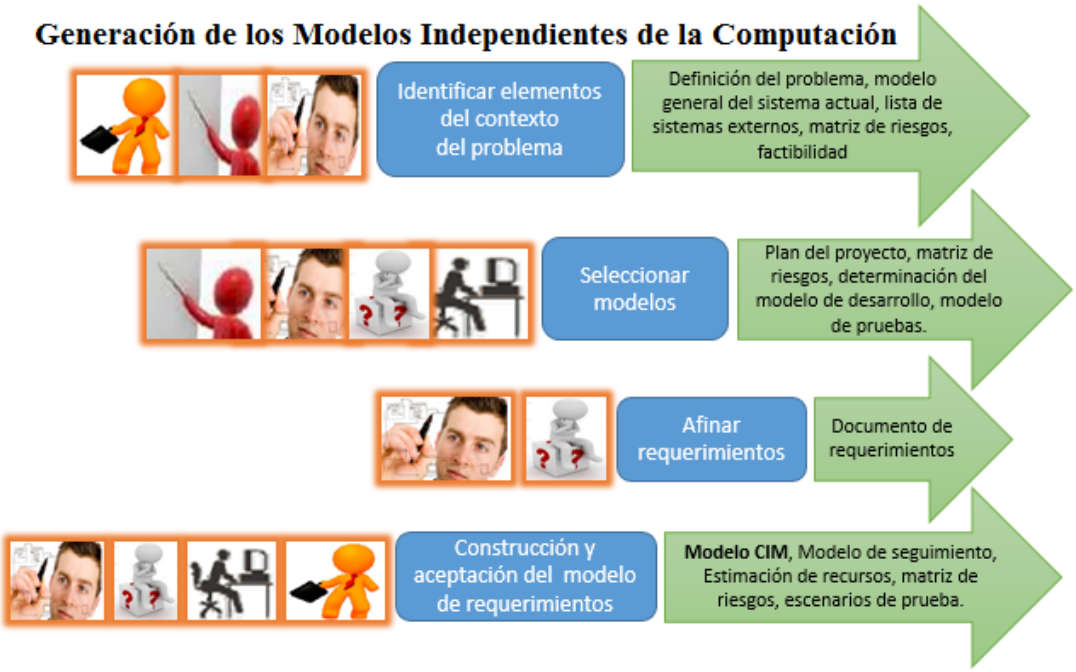


Figura 4.2: Recursos, Actividades y Salidas para la construcción de CIM´S, cumpliendo con lineamientos de CMMI Fuente. Propia.

- **Seleccionar modelos**: Dependiendo del tipo de aplicación a desarrollar, el tamaño del sistema, y los riesgos del desarrollo se seleccionará el modelo de desarrollo, se elaborará la documentación de la viabilidad técnica, el Gerente del proyecto elaborará el Plan del proyecto, el arquitecto y el analista elaborarán el modelo de seguimiento o trazabilidad, el probador definirá el modelo de pruebas.

- **Afinar requerimientos**: Partiendo de la definición del Plan del proyecto, el modelo de procesos, y el documento de requerimientos, el arquitecto y analista construyen el modelo CIM, el Gerente del proyecto afinará el alcance, la estimación de recursos y la matriz de riesgos y el probador definirá los escenarios de pruebas.

- **Construcción del modelo de requerimientos**: En esta actividad el arquitecto y el analista crearán las instancias de los requerimientos, procesos de negocio y modelo de dominio. La creación de los casos de uso podrían hacerse manualmente o a través de transformaciones. El probador podría hacer la validación de los requerimientos y revisar las transformaciones que se hayan hecho. Por último los stakeholders deberían aprobar el modelo de requerimientos y el Plan del proyecto.

Esta actividad facilita la elaboración de los CIM´s definitivos.

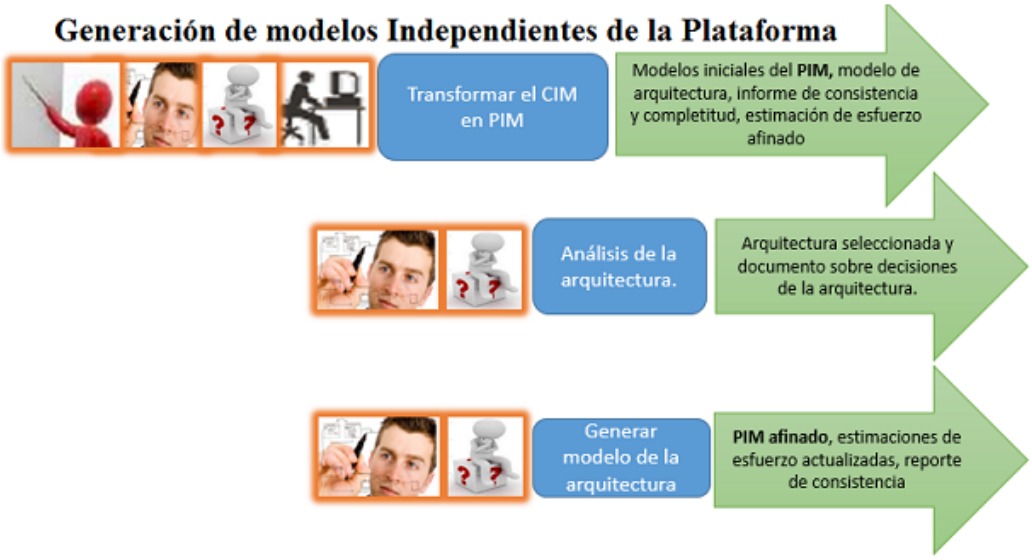


Figura 4.3: Recursos, Actividades y Salidas para la construcción de PIM´S, cumpliendo con lineamientos de CMMI. Fuente. Propia.

4.2. Generación de los Modelos Independientes de la Plataforma

Esta etapa del desarrollo MDA, se identifica con las fases de análisis y diseño de la solución. Los modelos PIM se constituyen por elementos como diagramas de uso, diagramas de actividades,y diagramas de clases entre otros que presentan la dinámica del sistema.

La gráfica 4.3 muestra el proceso de Generación de los Modelos Independientes de la Programación, o PMI´s. :

- **Transformar CIM**: El analista y el arquitecto generan la primera versión del PIM. Además, se hacen las correcciones o adiciones a los modelos PIM. A partir de los PIM´s generados se evalúan nuevamente los riesgos. En ésta etapa se reutilizan los patrones de análisis. El probador deberá verificar la completitud de los modelos y los escenarios de prueba.

- **Análisis de la arquitectura**: El arquitecto determinará cuál es la arquitectura apropiada para el sistema de acuerdo al tipo de aplicación. La arquitectura afectará la aplicación de patrones que permitirán refinar el modelo PIM.

- **Generar modelos PIM**: En esta etapa el arquitecto y el analista consideran la reutilización de componentes arquitectónicos predefinidos, hacen las transformaciones necesarias para generar los PSM´s. El probador verificará la consistencia y completitud de los modelos CIM y PIM´s.

  Esta actividad facilita la elaboración de los PIM´s definitivos.

Figura 4.4: Recursos, Actividades y Salidas para la construcción de PSM´S, cumpliendo con lineamientos de CMMI. Fuente. Propia.

4.3. Generación de los Modelos de Plataforma Específica - PSM

Los modelos PSM se constituyen por elementos como clases en JAVA o C++, esquemas de bases de datos, y asociaciones de intercambio de información.

La gráfica 4.4 muestra el proceso de Generación de los Modelos de Plataforma Específica - PSM´s. :

- **Generar los modelos de diseño PSM´s**: La generación de modelos PSM´s facilita la verificación de la consistencia y completitud del Sistema. Con base en estos modelos se evalúan nuevamente los riesgos y se deja todo listo para la realización de las pruebas.

4.4. Generación de Código

En esta actividad el Arquitecto, y el desarrollador utilizan las herramientas y reglas de transformación para generar el código básico de la aplicación

que se desarrolla.

Figura 4.5: Recursos, Actividades y Salidas para generación del código básico. Fuente. Propia.

La gráfica 4.5 muestra el proceso de transformación de modelos PSM´s en código. :

- **Generar el código básico**: En esta actividad el Arquitecto, y el desarrollador utilizan las herramientas y reglas de transformación para generar el código básico de la aplicación que se desarrolla.

A continuación para tener un cuadro más completo de los procesos en un desarrollo MDA, se analizan las funciones de cada rol involucrado en el desarrollo según Mellor S. y Watson A. (5).

4.5. Roles de hoy en el Proceso de MDA

Es necesario que el equipo que conforma el equipo del proyecto de desarrollo tenga claro su participación en cada uno de los procesos, con el fin de mejorar su productividad y disminuir los riesgos.

En un proyecto de gran magnitud, y en una empresa donde se dispone del recurso, las actividades pueden ser realizadas por personas diferentes. En otras situaciones una misma persona podría realizar diferentes roles siempre y cuando tenga el conocimiento necesario, a excepción del rol de probador, que es incompatible con los demás roles, debido a razones de independencia. Además, cuando se realizan fusiones de varios roles en una misma persona debe tenerse en cuenta que generalmente las preocupaciones de cada actor son diferentes; por ejemplo, las preocupaciones del analista no son las del programador.

De acuerdo al orden en que intervienen en los procesos son los siguientes: (5)

Patrocinador, stakeholders, usuarios y clientes: Son supremanente importantes en las etapas de requerimientos y validación. En el desarrollo con MDA continúan teniendo la misma importancia que en cualquier desarrollo, con el fin de identificar el contexto del problema, definir y validar sus necesidades, el modelo de requerimientos y el Plan del proyecto. Pero en ningún momento deberán influir en las estimaciones de recursos que haga el Gerente del Proyecto en cuanto a tiempo y talento humano necesario para el desarrollo e implementación, pues no tienen el conocimiento técnico que se requiere para esta actividad.

Arquitectos: Deciden sobre la estructura general del sistema, y van a seguir haciéndolo en MDA. En un proceso MDA, este trabajo implicará la participación en las actividades de identificación del contexto del problema, selección de los modelos, afinación de requerimientos, y costrucción del modelo de requerimientos, transformación del CIM en PIM, análisis de la arquitectura, generación del modelo de arquitectura, utilización de patrones, reutilización de componentes arquitectónicos predefinidos, transformación del PIM en PSM, y generación del código.

Si el mercado es maduro, los arquitectos encontrarán modelos y asignaciones existentes para su reutilización. Los Arquitectos aplican su experiencia en el proceso de selección y en los ajustes para mejorar el rendimiento del sistema.

Analista de requerimientos: La tarea de reunir y resolver requerimientos no cambia. El analista de requerimientos prepara los resultados del análisis de los requerimientos como modelos UML que pueden ser leídos por la máquina, y que pueden ser utilizados para probar y verificar los modelos construidos más adelante en el proceso de desarrollo; ésto reduce las posibilidades de que los resultados del análisis de los requerimientos serán mal interpretadas.

Participa en las actividades de seleccionar modelos, afinar requerimientos, construir del modelo de requerimientos, validar y verificar de requerimientos. Los modelos UML como diagramas de actividad y diagramas de casos de uso sin duda pueden ayudar en la

comunicación y la visualización de la requerimientos, pero su creación no puede ser automatizada por el momento.

Analistas diseñadores Es especialmente importante que los diagramas de clases, diagramas tabla de estado y métodos tengan un alto nivel de exactitud debido a que serán transformados en modelos.

Los analistas diseñadores deben decidir si es pertinente utilizar patrones de diseño en la aplicación que están desarrollando, o adicionar componentes mediante la ejecución de reglas de transformación.

Analista / programadores Las necesidad de habilidades de los programadores son las mismas que antes, pero con dos diferencias. En primer lugar el idioma en el que expresan sus abstracciones será el definido por QVT, en lugar de Java C ++ o Smalltalk. En segundo lugar, las abstracciones que estos desarrolladores crean tienen un mayor apalancamiento. Su trabajo se convierte en la creación y puesta a punto de los patrones para generación del código. De esta manera la experiencia del analista - programador es utilizada mucho mejor que simplemente para escribir código a mano.

Los Programadores al trabajar con herramientas MDA descubren que pueden escribir el código como parte de una plantilla de generación de código, dándoles más tiempo para trabajar en las piezas individuales más interesantes de la lógica de que el código de un generador de código. Como consecuencia de ello, las herramientas menos avanzadas de MDA pueden generar por lo menos 50 % de todo el sistema, y las herramientas más sofisticadas puede hacer hasta el 100 % en ciertas circunstancias.

"La automatización propicia un aumento de la creatividad que los desarrolladores pueden aplicar; lo que obliga a que ellos piensen en un nivel de abstracción por encima de los lenguajes de programación o la tecnología de moda". (20)

Probadores y/o Auditores de Sistemas: Llegan a ser más productivos mediante el uso de herramientas para generar scripts de prueba a partir de modelos, que pueden probar las partes codificadas a mano y generadas automáticamente.

Al trabajar con una herramienta de prueba de última generación que genera código de prueba de un modelo, un probador puede producir rápidamente el gran número de pruebas individuales necesario para probar exhaustivamente una aplicación grande, pero

sin el tedio de la escritura y reescritura del código.
El probador deberá definir el modelo de pruebas, Ser actor principal en la ejecución de las pruebas junto con los usuarios, validar el modelo de requerimientos, verificar la completitud de los modelos, aprobar los ambientes de prueba. El auditor de Sistemas estará permanentemente evaluando los riesgos.

Personas de Mantenimiento El mantenimiento de una aplicación MDA es una menos frustrante y más productiva que la actividad de mantenimiento de código escrito a mano debido a que su documentación es más legible y actualizada que el del código a mano. El personal de mantenimiento deben mantener los modelos y el diseño reglas, no el código.

4.5.1. Resumen y conclusiones del capítulo

Luego de revisar en el capítulo 3 los diferentes modelos que se crean en un desarrollo MDD, y los riesgos en su construcción y procesos de transformación, el éste capítulo fué necesario ahondar más en las actividades y el equipo de desarrollo requerído en cada una de las etapas de desarrollo, específicamente bajo el modelo MDA, con el objetivo de buscar otros riesgos o fortalezas en el paradigma de desarrollo, derivados de las actividades y el personal, para llegar finalmente a plantear los artefactos a implementar, buscando ser 100 % consistentes con los lineamientos en los niveles 2 y 3 del modelo CMMI.

En el desarrollo de éste capítulo, se identificó la importancia que tiene el arquitecto de software y su alta participación en los desarrollos bajo MDA. Ver tabla 5.1. Además, de detectó la necesidad de evaluar los riesgos luego de cada transformación de los modelos CIM, PIM y PSM.

Capítulo 5

Interpretación de CMMI® para Desarrollo, Versión 1.3 en enfoques Dirigidos por Modelos

"Las fortalezas están en las diferencias, no en las similitudes".

Stephen Covey.

Teniendo en cuenta los riesgos propios del desarrollo utilizando el enfoque MDD, y las tareas de cada uno de los miembros del equipo de desarrollo, clientes y usuarios, se presenta en éste capítulo una propuesta del encuentro entre los dos modelos MDD y CMMI.

El objetivo de este capítulo es indicar cómo el desarrollo por modelos cumple con lineamientos del modelo CMMI, y cuáles podrían ser las revisiones que un evaluador de CMMI debería realizar cuando evalúa procesos de los niveles 2 y 3 en empresas que utilizan el enfoque MDD.

Para ello se ha revisado la documentación acerca de la Interpretación de CMMI® para Desarrollo, en enfoques ágiles (19), pretendiendo proponer de forma similar las notas que permitan interpretar las prácticas CMMI a los responsables de desarrollo de software que utilizan MDD, y específicamente MDA.

Se incluye el análisis de las siguientes áreas de proceso de los niveles 2 y 3.

Nivel 2

- CM Gestión de la Configuración
- PMC Seguimiento y Control de Proyecto
- PP Planificación de Proyecto
- PPQA Aseguramiento de la Calidad de Proceso y Producto
- REQM Gestión de Requerimientos

Nivel 3

- PI Integración de Producto
- RD Desarrollo de Requerimientos
- RSKM Gestión de Riesgos
- TS Solución Técnica
- VER Verificación.

La presentación de los resultados del análisis de cada una de las prácticas seguirá la siguiente estructura:

- Propósito de la practica según CMMI.
- Problemas detectados.
- Solución propuesta.

5.1. Directrices para Áreas de proceso CMMI Nivel 2

5.1.1. CM Gestión de la Configuración

Propósito de la practica según CMMI: "Establecer y mantener la integridad de los productos de trabajo utilizando la identificación de la configuración, el control de la configuración, el informe del estado de la configuración y las auditorías de la configuración."(1).

Tomando la definición de [Pressman 2006], la Gestión de la configuración

se encarga de la identificación, organización, y control de la evolución de los requerimientos del software cuando cambian debido a la variabilidad de las reglas del negocio, la re-ingeniería del software, o las decisiones que soportan la arquitectura del sistema.

Problemas detectados : En el desarrollo dirigido por modelos es importante establecer un marco de gestión de la configuración que ayude a controlar la transformación de modelos y facilite la verificación y la validación de la exactitud de los requerimientos.

Solución propuesta : Se hace necesario fortalecer las prácticas de Gestión de la configuración y utilizar algo más que plantillas para formalizar las solicitudes de cambio. Utilizar los modelos de trazado a partir de metamodelos, o modelos de lenguajes de modelado, facilita la utilización de patrones dinámicos que estandarizan y guían la práctica de la trazabilidad a partir de los criterios de calidad.

Definir modelos de trazabilidad como controladores de la transformación de los modelos en el contexto MDD, mejora la gestión de la configuración en servicios como: la verificación de la consistencia y la completitud, y el manejo del cambio.(18)

En resumen, en el desarrollo dirigido por modelos se deberían establecer los Modelos de Trazado como el marco conceptual que soporten la gestión de la configuración

5.1.2. PMC - Seguimiento y Control de Proyecto

Propósito de la practica según CMMI: "Proporcionar una comprensión del progreso del proyecto para que se puedan tomar las acciones correctivas apropiadas, cuando el rendimiento del proyecto se desvíe significativamente del plan"(1).

Problemas detectados : Analizar y regular el progreso y el desempeño del proyecto, sirve para identificar áreas en las que el plan requiera cambios y se deban realizar los cambios correspondientes de manera oportuna. Una ventaja del desarrollo por modelos, es que el líder del proyecto obtiene datos del progreso del proyecto basado en el uso de herramientas MDD, en lugar de basarse en estimaciones del programador.(20)

El Control de Riesgos y la gestión del recurso humano es especialmente importante en el desarrollo dirigido por modelos, debido a los riesgos que involucra. Como se mencionó en el capítulo 3 de éste trabajo, un riesgo en MDD, es que no se incluyan completamente todos los actores en el modelo.

Cada actor realiza actividades críticas, puntuales y diferentes que difícilmente podrían ser ejecutadas por otro actor.

Solución propuesta : A continuación se muestra una tabla resumen de las actividades y el actor correspondiente, con el fin de tener en cuenta la inclusión de un responsable para cada una de las actividades involucradas en el desarrollo bajo MDD. se recomienda que el Arquitecto participe a lo largo de todo el proceso de desarrollo, como se ilustra en la tabla 5.1.

Tabla 5.1: Actividades por Rol en desarrollo dirigido por modelos

Actividad/rol	**1.**	**2.**	**3.**	**4.**	**5.**	**6.**	**7.**
Identificar elementos del contexto del problema	X	X	X				
Seleccionar modelos		X	X	X			X
Afinar requerimientos			X	X			
Construcción y prueba del Modelo de Requerimientos	X		X	X			X
Transformar CIM en PIM y verificar completitud	X		X	X			X
Analizar la arquitectura			X	X			
Generar modelos PIM			X	X			
Generar los modelos de diseño PSM´s		X	X	X	X		X
Generar el código básico	X	X	X			X	X

Convención: Rol:

- 1. Patrocinador, stakeholders, usuarios y clientes.
- 2. Líder del proyecto.

- 3. Arquitecto.
- 4. Analista de requerimientos.
- 5. Analistas diseñadores.
- 6. Analista / programadores.
- 7. Probadores y/o Auditores de Sistemas.

5.1.3. PP - Planificación del Proyecto

Propósito de la practica según CMMI: "Establecer y mantener planes que definan las actividades del proyecto"(1).

Problemas detectados : En los desarrollos dirigidos por modelos es especialmente importante planificar el alcance, la estimación de recursos y la matriz de riesgos.

Solución propuesta : El líder del proyecto, el Arquitecto y el Analista de Requerimientos deberán asegurarse de que se incluyan todas las reglas del negocio requeridas para el modelo del proceso de negocio que se modela y que este modelo sea verificado y validado por el probador y los usuarios.

Es necesario además, y utilizar técnicas y tecnologías que soporten la construcción de modelos sin ambigüedades. Por ejemplo, para la construcción del CIM es recomendable el uso de un lenguaje gráfico notacional que permita la construcción de modelos más precisos que los construidos usando UML.

5.1.4. PPQA - Aseguramiento de la Calidad de Proceso y Producto

Propósito de la practica según CMMI: "Proporcionar al personal y a la gerencia una visión objetiva de los procesos y de los productos de trabajo asociados"(1).

Problemas detectados : La trazabilidad de requerimientos es un atributo de calidad tratado por la ingeniería de software, y en el desarrollo por Modelos, se requiere un buen control de la consistencia y completitud de los requerimientos para que éstos no sean alterados con la continua transformación de los modelos.

Solución propuesta : El modelo de trazado asumido en el proyecto de desarrollo y definido en el plan de calidad de la empresa son herramientas recomendadas de aseguramiento de la calidad del producto y del proceso.

5.1.5. REQM - Gestión de Requerimientos

Propósito de la practica según CMMI: "Gestionar los requisitos de los productos y los componentes de producto del proyecto, y asegurar la alineación entre esos requisitos, y los planes y los productos de trabajo del proyecto" (1).

Problemas detectados : La Gestión de los requerimientos puede verse afectada en MDD por factores como la complejidad de los modelos, cambios invasivos no controlados, modelos parcialmente actualizados cuando hay cambios, matrices de trazabilidad no actualizadas, y altos costo para la realización de esta práctica.

Solución propuesta : Se recomienda el uso de prácticas como mantener la trazabilidad bidireccional mediante el uso de herramientas de gestión del trabajo y de configuración.

5.2. Directrices para áreas de proceso CMMI nivel 3

5.2.1. PI - Integración de Producto

Propósito de la practica según CMMI: "Ensamblar el producto a partir de sus componentes, asegurar que el producto, una vez integrado, se comporte correctamente (es decir, que posea la funcionalidad y los atributos de calidad requeridos). "(1).

Problemas detectados : En el desarrollo basado en modelos, el producto es construido a partir de componentes heterogéneos, ignorando las diferencias de implementación entre los componentes o aplicaciones. Por ésta razón puede decirse que MDD atiende por naturaleza el asunto de la integración.

Solución propuesta : Se recomienda tener un mecanismo para certificar que los artefactos y modelos cumplan los estándares y se mantenga la integridad del sistema.

5.2.2. RD - Desarrollo de Requerimientos

Propósito de la practica según CMMI: "Obtener, analizar y establecer los requisitos de cliente, de producto y de los componentes de producto. "(1).

Problemas detectados : En un desarrollo MDD, es posible que las actividades de modelado inicien o no en el nivel de requerimientos. Cuando el modelado inicia en los requerimientos, se construye en primera instancia el modelo Independiente de la Computación, CIM, que contiene las reglas de negocio y los requerimientos del sistema (modelos de negocio y modelos de dominio). Sin embargo, si no es así, o si la transformación CIM-PIM no se hace de manera automática, es posible que se incrementen los riesgos y el CIM no ayude a tener unos requerimientos precisos, claros y completos.

Solución propuesta : Existen varios estándares y herramientas que pueden utilizarse para la construcción del CIM que proporcionan una serie de elementos para representar de manera estandarizada los métodos, ciclos de vida, roles, actividades, tareas y productos de trabajo utilizados en Ingeniería de software. Se recomienda utilizar herramientas que faciliten la transformación automática de CIM a PIM.

En los casos en que el equipo decida iniciar la construcción de modelos posterior a la etapa de refinamiento de requerimientos, el equipo deberá definir los artefactos que serán los ejes de trazado de requerimientos.

5.2.3. RSKM - Gestión de Riesgos

Propósito de la practica según CMMI: "Identificar problemas potenciales antes de que ocurran, para que las actividades de tratamiento de riesgos puedan planificarse e invocarse según sea necesario a lo largo de la vida del producto o del proyecto para mitigar los impactos adversos sobre la consecución de objetivos. "(1).

Problemas detectados : En MDD se hace una constante gestión del riesgo a través de las actividades de definición y aplicación de escenarios de prueba, ajustes y afinación de modelos. Sin emabargo, no hay que descuidar ésta área de proceso.

Solución propuesta : Se recomienda que durante la actividad de identificación del contexto del problema, se elabore la matriz de riesgos vs actividad de desarrollo.

Deberían realizarse pruebas durante las siguientes actividades y considerar los correctivos necesarios:

- Construcción del modelo de requerimientos: Validaciones de completitud y consistencia.
- Transformación CIM a PIM.

- Generación de modelos de arquitectura.
- Transformación a código.

Cabe anotar que en MDD la transformación controlada por los modelos de trazado ayudan a disminuir los riesgos al reducir la subjetividad en las decisiones de modelado.

5.2.4. TS - Solución Técnica

Propósito de la practica según CMMI: "Seleccionar, diseñar e implementar soluciones para los requisitos. Las soluciones, los diseños y las implementaciones engloban productos, componentes de producto y procesos del ciclo de vida relativos al producto, bien individualmente o en conjunto, según proceda. "(1).

Problemas detectados : El reto es aprovechar al máximo la reutilización de herramientas y componentes construidos en proyectos de desarrollo anteriores.

Solución propuesta : "La selección de los componentes a reutilizar la hará el arquitecto comparando los patrones comunes identificados, con los activos MDD existentes. Los activos MDD existentes, provienen de proyectos MDD previos y/o de paquetes y herramientas estándar. Además, el arquitecto de soluciones deberá elegir el tipo de modelo apropiado, (UML u otro), para que utilicen los desarrolladores de la aplicación También como una actividad en esta área, se definen las herramientas MDD necesarias para desarrollar el proyecto." (20)

5.2.5. VER - Verificación

Propósito de la practica según CMMI: "Asegurar que los productos de trabajo seleccionados cumplen los requisitos especificados."(1).

Problemas detectados : En MDD es necesario realizar verificación tanto del framework del modelo como de todos los artefactos solución generados (21).

En cada una de las actividades de transformación de modelos se debe realizar la verificación entre el artefacto construido y los requerimientos, y cada herramienta MDD desarrollada debe ser probada apropiadamente.

Solución propuesta : Es recomendable que los casos de prueba puedan

ser escritos para que registren automáticamente los resultados de las pruebas cada vez que se ejecuten.

5.3. Conclusiones sobre las directrices presentadas

En la siguiente tabla se muestra un resúmen de los resultados obtenidos en la investigación.
Estas recomendaciones son de utilidad al equipo de desarrollo de software y al evaluador de los procesos CMMI, con el fin de asegurar y revisar el cumplimiento de los procesos de los niveles 2 y 3 en empresas que utilizan el enfoque de desarrollo MDD.

Tabla 5.2: Resumen de recomendaciones para el cumplimiento de las prácticas CMMI niveles 2 y 3 en un desarrollo bajo MDD. Fuente. Propia.

Área de proceso	**Problema**	**Artefacto propuesto**
Gestión de la Configuración (CM).	Se debe controlar la transformación de modelos además de la validez y verificación de los requerimientos.	Utilizar los modelos de trazado a partir de metamodelos, o modelos de lenguajes de modelado.
Seguimiento y Control del Proyecto (PMC).	Que no se incluyan completamente todos los actores en el modelo.	Cada uno de los roles del equipo de desarrollo debe tener un responsable para garantizar la ejecucuión de todas las actividades.
PP - Planifcación de Proyecto.	En los desarrollos dirigidos por modelos es especialmente importante planificar el alcance, la estimación de recursos y la matriz de riesgos.	Incluir todas las reglas del negocio requeridas para modelar el proceso en cuestión y que este modelo sea verifcado y validado por el probador y los usuarios.
PPQA - Aseguramiento de la Calidad de Proceso y Producto.	Los requerimientos podrian ser modificados en el proceso de transformación de los modelos.	Definir el modelo de trazabilidad de los requerimientos y un plan de calidad.

Continua...

Tabla 5.2 – Continuación...

REQM - Gestión de Requerimientos.	La complejidad de los modelos, los cambios no controlados, los modelos parcialmente actualizados cuando hay cambios, y las matrices de trazabilidad no actualizadas, pueden afectar el cumplimiento de los requerimientos.	Utilizar los modelos de trazado a partir de metamodelos, o modelos de lenguajes de modelado.
PI - Integración de Producto.	MDD atiende por naturaleza el asunto de la integración.	Sin embargo, Se recomienda tener un mecanismo para certificar que los artefactos y modelos cumplan los estándares y se mantenga la integridad del sistema
RD - Desarrollo de Requerimientos.	Que los requerimientos se pierdan o sean modificados en las transformaciones de modelos.	Se deberán definir los artefactos que serán los ejes de trazado de requerimientos.
RSKM - Gestión de Riesgos	En los desarrollos MDD, al igual que en cualquier desarrollo debe realizarse un constante análisis de riesgos.	Se recomienda de que durante la actividad de identificación del contexto del problema se elabore la matriz de riesgos vs actividad de desarrollo y se realicen pruebas luego de cada transformación de modelos.
TS - Solución Técnica	Se debe atender la reutilización de componentes y determinar cuáles son las herramientas MDD necesarias para el desarrollo del proyecto, con el fin de utilizar las ventajas que ésta práctica conlleva.	El arquitecto debe definir los artefactos MDD a reutilizar.

Continua...

Tabla 5.2 – Continuación...

VER - Verifcación	Se deberá facilitar la verificación tanto del framework del modelo como de todos los artefactos solución generados.	Se recomienda escribir cada caso de prueba para generar automaticamente los resultados de las pruebas.

Capítulo 6

Riesgos Propios de las Herramientas Actuales

"El miedo a lo desconocido es un potente inhibidor de la adopción de una nueva tecnología".

Este capítulo aborda el análisis de riesgos que no son propios al proceso de desarrollo por modelos, pero que podrían afectar el proceso y la calidad del producto si no son mitigados oportuna y adecuadamente.

Se consideró dedicar un capítulo separado al análisis de estos riesgos debido a que no son inherentes al proceso, y tenderán a desaparecer en la medida en que evolucionen las herramientas y los desarrolladores se familiarizan con el paradigma de desarrollo por modelos.

Se analizaron varios documentos tratando de abarcar diferentes percepciones:

- **Autores familiarizados con CMMI**: "Model-Driven Engineering: Automatic Code Generation and Beyond" (basado en un trabajo financiado y apoyado por el Departamento de Defensa bajo el contrato No. FA8721-05-C-0003 con la Universidad de Carnegie Mellon) , publicado en el 2015 (23),

- **CMU/SEI-2005-TN-022**: "Model Problems in Technologies for Interoperability: Model-Driven Architecture", 2005 (27)
- **Autores de la Academia**: "Desarrollo de software dirigido por modelos", 2012" (24).
- **Autores investigadores de MDD**: "Análisis y Evaluación del MDD (Model Driven software Development) desde la Perspectiva del Nivel 2 del CMMIDEV 1.3" del 2012.(21)
- **Empresas constructoras de herramientas MDD**: "The leading platform for Model Driven Architecture (MDA)", ARCSTYLER 2005.(26)

6.1. Actuales retos de MDD y MDA

- **Costos iniciales en la adopción de las técnicas y herramientas MDD debido a la curva de aprendizaje e inversión que representan para cualquier organización**:

  - En lo que se refiere al costo de las herramientas, éste debería retornarse con el tiempo en la medida que se construyen los repositorios y el reuso se hace factible.

    Además, existen herramientas abiertas que pueden considerarse por lo menos como una primera opción para las empresas que innovan con este paradigma de desarrollo. En la actualidad no se encuentra mucha documentación sobre costos de desarrollo MDD. Sin embargo, según ArcStyler, el uso de MDD se traduce en un ahorro inmediato global de más del 34 % en comparación con otros enfoques de desarrollo.(26)

    Según (23), los costos de desarrollo al utilizar MDD podrían incrementarse si no se considera al alineación entre las herramientas y la estrategia de compras. Para analizar éste deseado acoplamiento entre las herramientas y las estrategias de compra, recomienda que se evalué la compra de las herramientas contestando las siguientes preguntas.

    - ¿Las herramientas soportan la metodología de desarrollo de software de los programadores?
    - ¿Las herramientas de generación de código son capaces de integrarse con otras herramientas de desarrollo y gestión que permitan medir y hacer seguimiento de los avances en el desarrollo?

- ¿La metodología de desarrollo seleccionado y sus herramientas asociadas estarán disponibles y compatibles para el ciclo de vida esperado del sistema?

La tabla 6.1 muestra algunos de los costos de herramientas MDD.

- En cuanto a la curva de aprendizaje, ésta se ve compensada ya que los desarrolladores tendrán luego más tiempo para dedicarle a cuestiones de un nivel de abstracción mayor que construir código.

Tabla 6.1: Costos de algunas herramientas MDD

Herramienta / Característica	**Precio US, Mayo del 2017**	**URL producto, downloads**
Adonis Community Edition	Libre	`http://www.es.adonis-community.com/download.html`
AndroMDa	Open Source	`https://www.andromda.org/`
Bizagi	Libre	`http://es.kioskea.net/download/descargar-22170-bizagi-bpm-suite`
Bonita BPM	Libre	`http://bonita-bpm-community-edition.softonic.com/`
Together Architect, de Borland	Perecio no disponible	`http://www.borland.com/Products/Requirements-Management/Together/`
Eclipse Modeling Framework (EMF), BPMN2 Modeler	Libre	`http://www.eclipse.org/modeling/emf/downloads/?`
ObjectiF	US 1.190	`http://www.microtool.de/en/objectif-model-driven-development/`
Enterprise Architect, de Sparx Systems	599 precio unidad	`http://www.sparxsystems.com.au/products/ea/purchase.html#BusinessSoftware`
IBM Rational Rose Modeler, de IBM.	5,082 Floating user	`ftp://public.dhe.ibm.com/software/rational/web/datasheets/rose_ds.pdf`
ArcStyler de Interactive Objects Software GmbH	Perecio no disponible	`http://arcstyler.software.informer.com/`

Problemas de adopción por parte de los actuales equipos de desarrollo, al tratarse de prácticas bastante diferentes a las tradicionales (24): Este riesgo se mitiga si en los casos en que el equipo de desarrollo y mantenimiento no estén familiarizados con la herramienta, se les brinda la capacitación y el apoyo para permitir a la organización adquirir los conocimientos y habilidades necesarias para desarrollar y mantener el software.

"Problemas a nivel organizativo, ya que muchos gestores no están dispuestos a apoyar inversiones cuyo beneficio a muy corto plazo no es trivial (aunque a largo plazo puedan mejorar significativamente la productividad de la empresa)" (24). Este riesgo disminuye en la medida en que se den más casos de éxito de desarrollo MDD y éstos sean documentados, puesto que los administradores ganarían más confianza en el desarrollo por modelos.

Falta de madurez en algunas de las herramientas MDD (editores de modelos, generadores de código, máquinas de transformación), principalmente debido a la reciente implantación de esta disciplina. La historia nos demuestra que la mejora de las herramientas es un proceso que toma tiempo, pero siempre se da.

Riesgos asociados con la disponibilidad continua de las herramientas y la compatibilidad de las nuevas versiones de las herramientas. (23). Este riesgo se mitigará cuando los fabricantes de herramientas estén dispuestos a cambiar sus herramientas para el "bien común", y se acojan a estándares y hagan posible la integración de herramientas. (27)

Falta de buenas prácticas y de procesos de desarrollo MDD, lo que no permite adoptar esta práctica de forma unificada. (24). Esta debilidad se podría atender al integrar modelos de calidad en el desarrollo de software como CMMI con los procesos MDD.
Por todas las anteriores razones, y porque MDD, ha surgido para ayudar a tratar un problema que aún existe, valdría la pena que las empresas desarrolladoras experimentaran con un proyecto concreto y controlado y evaluaran sus bondades y debilidades.

La tabla 6.2 muestra un resumen de los riesgos que no son propios del paradigma de desarrollo de software por modelos. Estos riesgos tienen tendencia

a minimizar con el surgimiento de herramientas más maduras y el afianzamiento del nuevo paradigma de desarrollo de software.

Tabla 6.2: Riesgos no propios al proceso de desarrollo por modelos, y recomendaciones para afrontarlos.

Riesgo	Recomendación
Alto costo de herramientas	Usar software libre para los primeros proyectos de desarrollo con MDD y cuando se decida comprar, tener en cuenta en selección la experiencia de los programadores con la herramienta y la compatibilidad de la nueva herramienta con el software existente.
Desconocimiento de las herramientas por parte de los desarrolladores	Incluir la capacitación de los desarrolladores dentro del primer proyecto a desarrollar.
Resistencia al cambio y temor a la inversión	Documentar los casos de éxito en la medida en que se den.
Inmadurez de las herramientas: Editores de modelos, generadores de código, máquinas de transformación	Actualizar las versiones en la medida que se liberen, acogerse a estándares y favorecer la integración.
Falta de buenas prácticas y de procesos de desarrollo MDD	Integrar modelos de calidad en el desarrollo de software como CMMI con los procesos MDD.

6.1.1. Resumen y conclusiones del capítulo

A pesar de que el capítulo 5 ya se presentaron los artefactos propuestos para implementar en desarrollos MDD, y ser consistentes con los lineamientos para los niveles 2 y 3 del CMMI-DEV, que era uno de los objetivos a cumplir en éte trabajo, es necesario analizar también los riesgos temporalmente válidos, debido a la inmadurez, costos de las herramientas y el estado aun incipiente del uso del paradigma de desarrollo.

Los riesgos expuestos, se superarán en la medida que evolucionen las herramientas y las empresas desarrolladoras experimenten más con el modelo de desarrollo y acepten los cambios.

Capítulo 7

Trabajos relacionados

"La mente que se abre a una nueva idea,
jamás volverá a su tamaño original".
Albert Einstein.

Algunos autores han abordado el tema de la utilización del modelo de calidad CMMI en proyectos de desarrollo utilizando el paradigma de desarrollo modelos dirigido por modelos:

- Esterkin, en (28) presenta los resultados de un análisis para responder a la pregunta si las prácticas MDD, soportan el nivel de madurez 2 del CMMI. Como resultado del mapeo encuentra que hay varias áreas de procesos y prácticas específicas no soportadas por MDD.

  En la presente investigación, se analizaron los resultados ya validados en el trabajo de Esterkin en lo que se refiere a cuáles prácticas de CMMI nivel 2 son soportadas por MDD, y se identificaron los artefactos que ayudarían al cumplimiento de todas las prácticas específicas de los niveles 2 y 3 de CMMI.

- Iñigo Garro, consultor y colaborador del SEI en el Quality Working Group, constituido para mejorar la calidad de las evaluaciones CMMI y las prácticas de la profesión, transcribe en (19), basándose en el informe técnico CMU/SEI-2010-TR-033 del Software Engineering, las directrices del modelo CMMI relacionadas con las Metodologías ágiles, y además, explica aspectos claves que permiten la coexistencia de

CMMI y los enfoques ágiles. Si bien este trabajo se enfoca sobre otro paradigma de desarrollo de software, es conveniente tenerlo en cuenta ya que brinda las bases para la definición del mapeo entre el CMMI y los distintos paradigmas de desarrollo.

- En (30) los autores muestran cómo un modelo de madurez desarrollado dentro del proyecto "Modelware" ayudó a varias compañías a adoptar el modelo de desarrollo MDD. El proyecto fue desarrollado en España durante los años 2002 al 2006 y definía 5 niveles de madurez que ubicaban a las empresas que adoptaban MDD en diferentes grados, tanto de seguimiento de prácticas MDD como de uso de artefactos MDD. En lugar de definir diferentes niveles de cumplimiento de prácticas MDD, ésta tesis propone adoptar un modelo de calidad reconocido, como es el CMMI, respondiendo a la necesidad de integrar el control de las prácticas específicas de desarrollo en este nuevo paradigma de desarrollo de software.

- En (31), J. Quintero y R. Anaya plantean los retos que aun afronta el MDD, como son las limitaciones de las herramientas y la falta de integración de los modelo BPM en las transformaciones de modelos. En el artículo propuesto se hace también un reconocimiento de los riesgos actuales con ambas falencias, pero se analiza y se plantea una forma de mitigar estos riesgos mediante controles presentados en las notas que se sugiere incluir en el modelo CMMI. Nuestro trabajo agrega un análisis detallado de las prácticas del CMMI y su conexión con MDD.

- A. Lins de Vasconcelos, G. Giachetti, B. Marín y O. Pastor, en (32) han definido un proceso de desarrollo de software basado en MDD, que va desde los requerimientos hasta el código final, que integra elementos del framework I* y la metodología GORE (Goal-Oriented Requirement Engineering) y que es compatible con CMMI-DEV. En este caso, los autores se han enfocado en la definición de un nuevo proceso de desarrollo de software, garantizando, por construcción, su compatibilidad con el modelo de madurez. Nuestro trabajo no se enfoca en ningún proceso en particular, sino en la metodología MDD en general, y por lo tanto pretende ser aplicable a cualquier proceso de desarrollo MDD.

- Esterkin y Pons, en (29) realizan un análisis de las buenas prácticas de MDD y determinan si estas soportan las guías del nivel 2 de CMMI, y en qué medida las soportan. En éste trabajo se complementó su análisis al identificar cuáles artefactos podrían utilizarse para dar cumplimiento a las prácticas no son soportadas por MDD.

7.0.1. Resumen y conclusiones del capítulo

En éste capítulo se han presentado algunos trabajos anteriores, de otros autores, relacionados con el tema de la presente investigación.

Cabe resaltar, la investigación realizada por Esterkin, en (28), ya que fue de mucha ayuda en el análisis de consistencia de las prácticas MDD, con el nivel de madurez 2 del CMMI. En el presente trabajo, se analizaron los niveles 2 y 3, y además de identificaron los artefactos para mitigar los riesgos de inconsistencia con lineamientos CMMI-DEV, quedando de ésta forma completo el análisis que se propuso realizar la autora, para responder a la pregunta "Cómo integrar el modelo CMMI al modelo de desarrollo de software MDD".

Capítulo 8

Validación de la Propuesta

"Siempre es un ejercicio interesante "sincronizar" modelos disitintos y buscar sinergias entre ellos".
Encuestado.

Con el fin de validar la propuesta presentada en éste trabajo, se realizó una encuesta cuyos resultados se presentan en éste capítulo.

Esta encuesta fue difundida en dos foros de CMMI, un foro sobre MDD, y autores de artículos, libros y tesis citados en este trabajo.

La encuesta se encuentra disponible en: `https://docs.google.com/forms/d/16NaZSCvG2LXON618UfdEMwfBCpO9h6BJrBEWx7yUoWQ/viewform`.

El tema de la propuesta de integración de los dos modelos también ha sido presentado por la tesista, en un artículo publicado en la revista seriada Ventana Informática No. 35 (jul-dic). Manizales (Colombia): Facultad de Ciencias e Ingeniería, Universidad de Manizales. p. 13-30. ISSN: 0123-9678. Bajo el nombre "CMMI-DEV en el desarrollo de software bajo el paradigma de Desarrollo Guiado por Modelos, MDD". ISSN: 0123-9678. Link:

http://revistasum.umanizales.edu.co/ojs/index.php/ventanainformatica/issue/view/156/showToc
Ver Anexo 1.

8.1. Consideraciones generales

La difusión de la encuesta se hizo de manera abierta para que fuese contestada por personas con conocimiento en CMMI y/o MDD, con el fin de tener una idea de qué proporción de los encuestados maneja cada uno de los modelos. Sin embargo, para el análisis de ciertas preguntas que lo requieren, solo te tuvieron en cuenta las respuestas de las personas que dijeron conocer ambos modelos.

Se plantearon 8 preguntas:

1. ¿En qué país trabaja?
2. ¿Cuál es su ocupación?
3. ¿Tiene usted conocimiento y/o experiencia en cuál de los siguiente temas: CMMI, MDD? Se consideraron respuestas como ambos, o ninguno.
4. ¿Cree usted que el modelo CMMI debe incorporar nuevos lineamientos al surgir nuevos paradigmas de desarrollo?
5. ¿Considera usted que la última versión de CMMI-DEV, incluye las guías suficientes para emplear en un proyecto de desarrollo bajo el paradigma MDD?
6. ¿Considera usted que el modelo MDD debería ser "complementado" en un proceso de desarrollo con los lineamientos del modelo CMMI?
7. ¿Está usted interesado en conocer cómo integrar los dos modelo MDD y CMMI en un proceso de desarrollo de software?
8. Escriba un comentario si lo desea.

La única dificultad encontrada durante la encuesta, fue la consecución de personas con el suficiente conocimiento en el paradigma de desarrollo MDD, que se sintieran seguros de dar su opinión.

8.2. Resultados de la consulta

Iniciamos el análisis con la pregunta que reduce la muestra, y nos deja solo las respuestas de los profesionales con conocimientos en ambos modelos, por considerar que estas personas son las que pueden aportar con mejor criterio a las preguntas 4, 5 y 6:

¿Tiene usted conocimiento y/o experiencia en cuál de los siguientes temas: CMMI, MDD?.

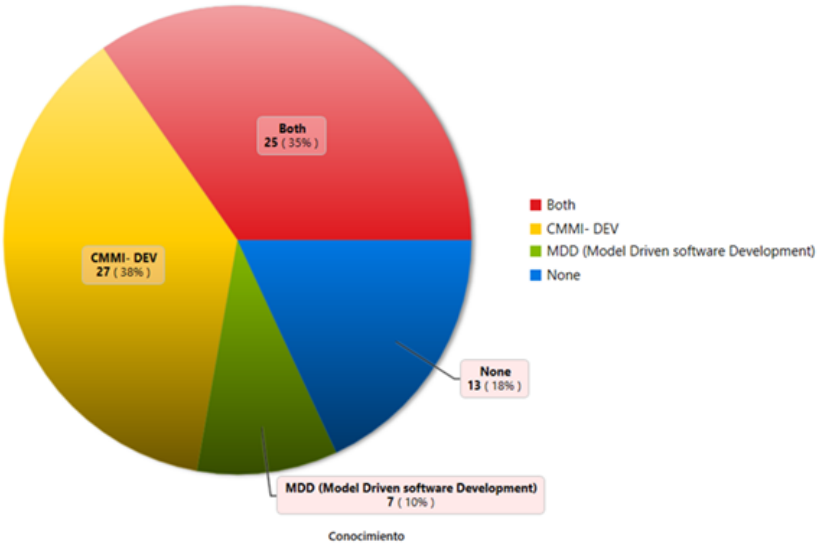


Figura 8.1: Medición de conocimiento en los modelos CMMI y MDD en las personas encuestadas.

El 35 % de las personas que respondieron la encuesta, aseguran tener conocimientos en CMMI y MDD: 25 de 72. Ver figura 8.1.

¿Considera usted que la última versión de CMMI-DEV, incluye las guías suficientes para emplear en un proyecto de desarrollo bajo el paradigma MDD?.

Para el análisis de la respuesta a ésta pregunta, solo se tuvo en cuenta a las personas que dijeron conocer ambos modelos. De las 25 personas que conocen ambos modelos, 18 (72 %), considera que las guías del modelo CMMI no son suficientes, o no saben si lo son. Esto podría considerarse un punto a favor de la validación de la propuesta, puesto que demuestra que las personas que saben de ambos temas han experimentado o saben sobre la falta de lineamientos específicos que harían falta en CMMI. Ver figura 8.2.

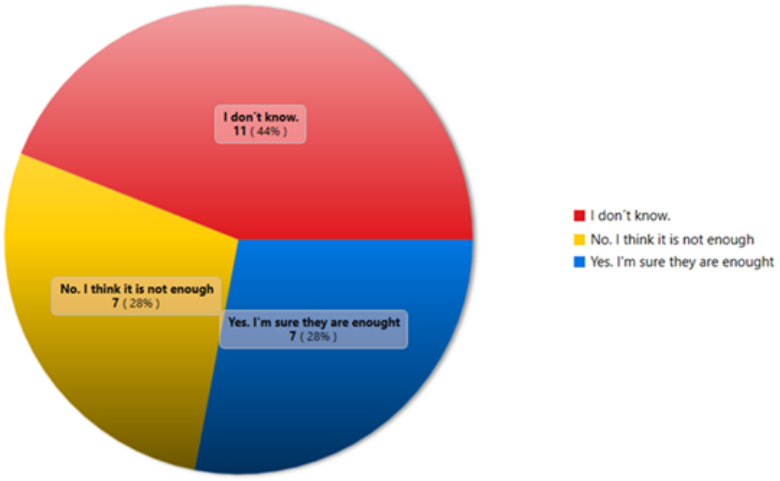


Figura 8.2: Opinión sobre la suficiencia del modelo CMMI.

¿Cree usted que el modelo CMMI debe incorporar nuevos lineamientos al surgir nuevos paradigmas de desarrollo?.

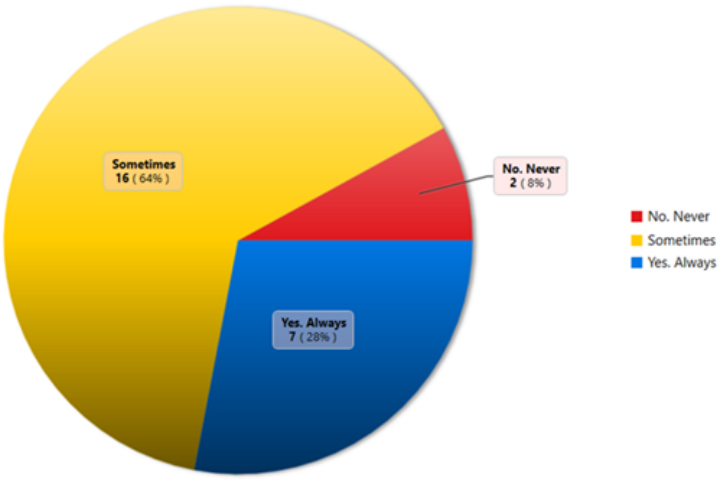


Figura 8.3: CMMI debería incorporar lineamientos sobre ante nuevos paradigmas?

El 92 % de las personas que conocen ambos modelos consideran que el modelo CMMI debe incorporar nuevas guías cuando surjan necesidades debido a nuevos paradigmas de desarrollo. Ver figura 8.3.

¿Está usted interesado en conocer cómo integrar los dos modelo MDD y CMMI en un proceso de desarrollo de software?.

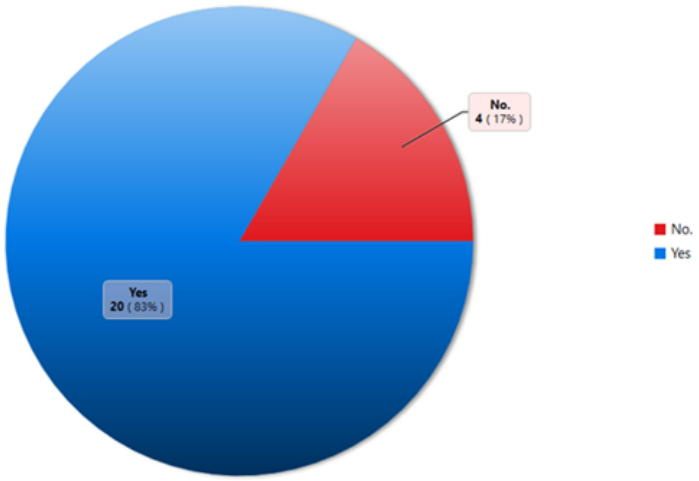


Figura 8.4: Interés en saber cómo integrar los dos modelos en un proyecto de desarrollo

.

El 83 % de las personas que conocen de ambos modelos manifestó interés en saber cómo integrarlos en un proyecto de desarrollo, lo que puede ser considerado como un concepto a favor de la propuesta. Ver figura 8.4.

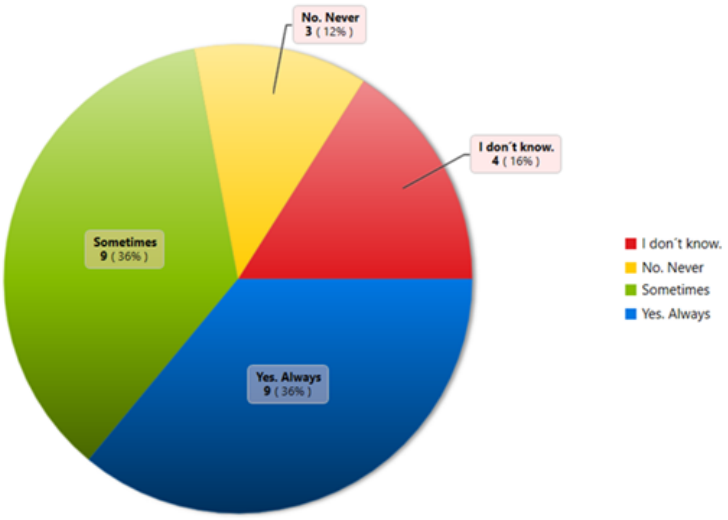


Figura 8.5: Deberían complementarse los modelos en un proyecto de desarrollo bajo MDD?

.

¿Considera usted que el modelo CMMI debería ser complementado en un proceso de desarrollo con los lineamientos del modelo CMMI?.

El 88 % de las personas que conocen de ambos modelos manifestó que ambos modelos deberían complementarse en un proyecto de desarrollo bajo el paradigma MDD, o no lo saben. Ver figura 8.5.

¿En qué país trabaja? y Conocimiento sobre los modelos CMMI y MDD?.

Las respuestas permiten sospechar sin tener en cuenta las diferencias en la población, que Estados Unidos, Argentina, y Colombia son los países que en estos momentos tiene más personas trabajando los modelos en cuestión. Debe tenerse en cuenta que la encuesta fue realizada solo para personas que hablan inglés y/o español. Ver figura 8.6.

¿Cuál es su ocupación? y Conocimiento sobre los modelos CMMI y MDD?.

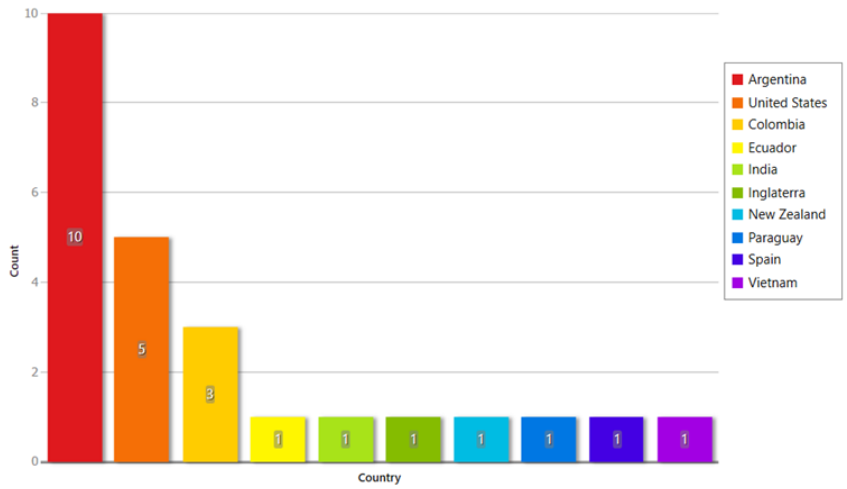


Figura 8.6: Datos cruzados sobre país donde labora el encuestado y conocimientos sobre los modelos

.

Tabla 8.1: Perfil de los encuestados. Fuente. Propia.

OCUPACIÖN	**Frecuencia**	**Porcentaje**
Chief Technology Officer	1	4 %
Consultant	1	4 %
EPG (Engineering Process Group) member	1	4 %
Executive management	1	4 %
Implementation advisor	1	4 %
Information technology consultant	1	4 %
Information technology consultant, Software developer, Project manager	1	4 %
Information technology consultant, University profesor	1	4 %
PMO, process improvement engineer	1	4 %

Continua...

Tabla 8.1 – Continuación...

Profesor e investigador	1	4 %
Project leader	1	4 %
Project manager	2	8 %
Project manager, University profesor	1	4 %
SCAMPI Lead Appraiser	1	4 %
Software developer	1	4 %
Software developer, Project manager, Software architect, University profesor, Researcher	1	4 %
Software developer, Project manager, Software architect, University profesor, Researcher, Other	2	8 %
Software developer, Researcher	1	4 %
Systems auditor	1	4 %
Systems engineer	1	4 %
University profesor	1	4 %
University profesor, Researcher	2	4 %
Total	**25**	**100 %**

Esta falta de conocimientos en el Desarrollo de Software Dirigido por Modelos puede verse como una debilidad para la integración de ambos modelos. Sin embargo, podría esperarse que esto se supere con el avance de herramientas MDD que faciliten el aprendizaje y la aplicación del paradigma de desarrollo MDD, como se indicó en el capítulo 6 de éste trabajo.

Analizando la ocupación de los encuestados, se aprecia que sería deseable que quienes trabajan más activamente en el desarrollo bajo MDD, como son los Arquitectoss, reportaran un mejor conocimiento del paradigma de desarrollo dirigido por modelos. Ver tabla 8.1.

8.2.1. Resumen de resultados de la consulta de validación

Tabla 8.2: Resumen de respuestas de la encuesta

Pregunta	Resultado
Conocimiento de ambos modelo CMMI y MDD	35 %
Suficiencia de CMMI-DEV en proyectos con MDD	72 % opinan que no, o no saben si es suficiente
Opinión sobre actualización del CMMI ante nuevos paradigmas de desarrollo	92 % consideran que sí debe actualizarse CMMI.
Interés en saber cómo utilizar CMMI en proyectos con MDD	83 % están interesados
Opinión sobre si el modelo CMMI debería ser complementado en un proyecto MDD	88 % consideran que sí deben complementarse
En qué paises según la encuesta, se un presenta mayor conocimiento de ambos modelos CMMI y MDD?	Argentina, Estados Unidos y Colombia
En qué tipo de profesión u oficio se muestra mayor conocimeinto de ambos modelos CMMI y MDD?	En los consultores TI, profesores e investigadores

El objetivo de la encuesta fue conocer la opinión de los expertos en ambos modelos, acerca de si las prácticas que se utilizan en un proyecto de desarrollo que utilice el paradigma MDD son suficientes para cumplir 100 % con las prácticas específicas definidas en CMMI DEV 1.3, en los niveles 2 y 3, o si es necesario la implementación de diferentes artefactos o enriquecer los que se aplican en un desarrollo tradicional, para su logro.

La encuesta fue respondida por 72 personas entre el 20 de mayo y el 16 de septiembre del 2015.

Al analizar las respuestas de manera conjunta, se puede apreciar que hay una aceptación e intrés en la integración de los modelos CMMI-DEV y MDD

en los proyectos proyectos desarrollo de doftware dirigido por modelos, en grupo de personas que manifestaron conocer ambos modelos. Además, los expertos consideran que no basta con los lineamientos dados para el desarrollo tradicional, para cumplir 100 % con los niveles 2 y 3 del CMMI-DEV 1.3. Ver Resumen de las respuestas en la tabla 8.2.

Capítulo 9

Conclusiones

En este trabajo se abordaron dos modelos importantes en el desarrollo de software: Uno con mucho reconocimiento y muchos casos de éxito documentados a nivel mundial, autoridad en el tema de la Calidad en el proceso y producto de software, y otro que a pesar de no estar maduro, promete darle un gran impulso a la industria del software, y cambiar el nivel de los procesos de desarrollo.

Su integración es necesaria puesto que no son antagónicos sino complementarios. Los que explotan el desarrollo por modelos necesitan de un estándar que defina los lineamientos y buenas prácticas del proceso de desarrollo, teniendo en cuenta los riesgos y particularidades del paradigma de desarrollo por modelos, y CMMI, debe cubrir el sector que será cada vez creciente de equipos de desarrollo y empresas que decidan incorporar nuevos paradigmas de desarrollo de software.

Sin embargo, para que esa integración se dé, es necesario que los investigadores, los fabricantes de herramientas MDD, la academia y el instituto CMMI, filial de Carnegie Innovations y quien es ahora responsable de todo lo relacionado con el entrenamiento, certificaciones, evaluaciones y mejora de procesos del modelo CMMI reúnan esfuerzos para interpretar las prácticas CMMI dentro de un proceso de desarrollo MDD. Esto incluiría por lo menos las siguientes actividades:

- Incorporar las notas necesarias en el modelo,
- Definir las prácticas a evaluar en cada proceso,
- Documentar y difundir las mejores prácticas como parte del modelo CMMI,
- Capacitar y certificar a los evaluadores de procesos den una empresa que desarrolle bajo MDD.

- Facilitar y motivar a los fabricantes de software para que estas herramientas "se hablen unas a otras".

Hacer realidad la integración, es una labor más compleja que no tiene que ver con el objetivo de este trabajo. Sin embargo, se ha podido comprobar según las respuestas obtenidas en la encuesta de validación, que existe la necesidad de contar con guías de calidad en proyectos de software bajo el paradigma de desarrollo por modelos, y que en la actualidad el modelo CMMI no cubre ciertos aspectos. Además, se logró identificar los riesgos inherentes al desarrollo por modelos, y cómo éstos podrían mitigarse incorporando mejores prácticas al modelo de calidad que se elija, y en especial al CMMI.

Bibliografía

[1] «CMMI® for Development, Version 1.3 CMMI-DEV, V1.3 - 10tr033.pdf». [En línea]. Disponible en: http://www.sei.cmu.edu/reports/10tr033.pdf. [Accedido: 22-jun-2014].

[2] «CMMI en México y el mundo (2015) | ::everac99». [En línea]. Disponible en: https://everac99.wordpress.com/2015/09/15/cmmi-en-mexico-y-el-mundo-2015/. [Accedido: 22-may-2017].

[3] «Maturity-Profile-Ending-March-2014.pdf». [En línea]. Disponible en: http://asprotech.blogspot.com.co/2016/03/actualizacion-del-perfil-cmmi-2015.html#more [Accedido: 22-jun-2014].

[4] «Edulp Editorial». [En línea]. Disponible en: http://libros.unlp.edu.ar/index.php/unlp/catalog/book/287.[Accedido: 25-may-2017]. se eliminó

[5] «Microsoft Word-Roles in MDA v6(clean).doc-Roles_in_MDA1.pdf». [En línea]. Disponible en: http://www.omg.org/registration/Roles_in_MDA1.pdf. [Accedido: 20-sep-2014].

[6] «BPMNbyExampleSPA.pdf». [En línea]. Disponible en: http://www.bizagi.com/docs/BPMNbyExampleSPA.pdf. [Accedido: 24-sep-2014].

[7] «Integración de modelos BPMN en ambientes MDA». [En línea]. Disponible en:http://sedici.unlp.edu.ar/handle/10915/23737. [Accedido: 24-sep-2014].

[8] «Una aproximación a la generación automática de código en un contexto MDD sobre modelos BPMN - 24_EST_2012.pdf». [En línea]. Disponible en:http://41jaiio.sadio.org.ar/sites/default/files/24_EST_2012.pdf. [Accedido: 30-sep-2014].

[9] «Microsoft Word - Art_12.doc - Art_12.pdf». [En línea]. Disponible en: http://ceur-ws.org/Vol-558/Art_12.pdf.[Accedido: 30-sep-2014].

[10] «Una visión del desarrollo de software utilizando modelos - Dialnet». [En línea]. Disponible en: http://dialnet.unirioja.es/servlet/articulo?codigo=4183290. [Accedido: 01-oct-2014].

[11] «Transforming software development: an MDA road map». [En línea]. Disponible en: http://ieeexplore.ieee.org/document/1510571/?reload=true [Accedido: 25-may-2017].

[12] «Ph.D. Iván Ruiz-Rube.pdf». [En línea]. Disponible en: http://rodin.uca.es/xmlui/bitstream/handle/10498/15725/Ph.D.%20Iv%C3%A1n%20Ruiz-Rube.pdf?sequence=1. [Accedido: 27-oct-2014].

[13] «Vector4_6.pdf». [En línea]. Disponible en: http://vector.ucaldas.edu.co/downloads/Vector4_6.pdf. [Accedido: 29-oct-2014].

[14] «Microsoft Word - article_malaysia[7] - transformation-method-cim-to-pim-from-business-processes-models-defined-in-bpmn-to-use-case-and-class-models-defined-in-uml». [En línea]. Disponible en: http://waset.org/publications/9999213/transformation-\method-cim-to-pim-from-business-processes\-models-defined-in-bpmn-to-use-case-and--models-defined-in-um. [Accedido: 28-oct-2014].

[15] «Microsoft Word - relais-v1-n4-142-145.doc - relais-v1-n4-p-142-146.pdf». [En línea]. Disponible en: http://sistemas.unla.edu.ar/sistemas/redisla/ReLAIS/relais-v1-n4-p-142-146.pdf. [Accedido: 05-ene-2015].

[16] «Process2008.pdf». [En línea]. Disponible en: https://www.risc.jku.at/people/schreine/papers/Process2008.pdf. [Accedido: 05-ene-2015].

[17] «Ingenieria de modelos con MDA - Ingenieria de modelos con MDA.pdf». [En línea]. Disponible en: http://dis.um.es/~jmolina/Ingenieria%20de%20modelos%20con%20MDA.pdf. [Accedido: 10-ene-2015].

[18] «Microsoft Word - Tesis vFinal - 42893042.2009_1.pdf». [En línea]. Disponible en: http://www.bdigital.unal.edu.co/2238/1/42893042.2009_1.pdf. [Accedido: 10-ene-2015].

[19] «Interpretacion de CMMI-DEV v1.3 en enfoques ágiles (web) - interpretacion-de-cmmi-dev-v1-3-en-enfoques-c3a1giles-web1.pdf». [En línea]. Disponible en: https://inigogarro.files.wordpress.com/2013/12/interpretacion-de-cmmi-dev-v1-3-en-enfoques-c3a1giles-web1.pdf. [Accedido: 04-mar-2015].

[20] «IBM Redbooks | Patterns: Model-Driven Development Using IBM Rational Software Architect». [En línea]. Disponible en: http://www.redbooks.ibm.com/abstracts/sg247105.html. [Accedido: 14-mar-2015].

[21] «Análisis y Evaluación del MDD (Model Driven software Development) desde la Perspectiva del Nivel 2 del CMMI-DEV 1.3». [En línea]. Disponible en:http://sedici.unlp.edu.ar/handle/10915/23705. [Accedido: 14-mar-2015].

[22] «The pragmatics of model-driven development - Software, IEEE - Selic-The Pragmatics of MDD.pdf». [En línea]. Disponible en: http://www.idt.mdh.se/kurser/ISD_cdt417/files/articles/byStudents/Selic-The%20Pragmatics%20of%20MDD.pdf. [Accedido: 26-mar-2015].

[23] "Model-Driven Engineering: Automatic Code Generation and Beyond - 2015_004_001_435420.pdf". [En línea]. Disponible en: http://resources.sei.cmu.edu/asset_files/technicalnote/2015_004_001_435420.pdf. [Accedido: 29-mar-2015].

[24] «Tecnicas_avanzadas_de_ingenieria_de_software_(Modulo_1).pdf». [En línea]. Disponible en: http://lasallelatacunga.edu.ec/repositorio/index.php/repositorio/Libros/Tecnolog%C3%ADa-Inform%C3%A1tica/Desarrollo-de-Software-Dirigido-por-Modelos/

[25] «sv-lncs - Documento_completo.pdf». [En línea]. Disponible en: http://sedici.unlp.edu.ar/handle/10915/23705 [Accedido: 25-may-2017]

[26] «PRODUCT - ArcStyler5_Whitepaper_220205.pdf». [En línea]. Disponible en: http://www.omg.org/mda/mda_files/ArcStyler5_Whitepaper_220205.pdf [Accedido: 01-abr-2015].

[27] «Model Problems in Technologies for Constructive Interoperability: Model-Driven Architecture - 2005_004_001_14510.pdf». [En línea]. Disponible en: http://resources.sei.cmu.edu/asset_files/TechnicalNote/2005_004_001_14510.pdf [Accedido: 04-abr-2015].

[28] V. Esterkin and C. Pons. «Análisis y Evaluación del MDD. Model Driven Development desde la Perspectiva del Nivel 2 del CMMI DEV 1.3». Publicado en proceedings of CACIC. Congreso Argentino de Ciencias de la Computación. Argentina. 2012.

[29] V. Esterkin. «Análisis y Evaluación del MDD (Model Driven Development) desde la perspectiva de CMMI DEV 1.3 Nivel 2». Tesis para optar al grado de Magíster. Universidad Abierta Interamericana. Ciudad de Buenos Aires, Argentina 2014.

[30] E. Ríos, T. Bozheva, A. Bediaga, N. Guilloreau. A. Rensink and J. Warmer. «MDD Maturity Model: A Roadmap for Introducing Model-Driven Development». A. Rensink and J. Warmer (Eds). ECMDA-FA 2006. LNCS 4066, pp 78-89. 2006.

[31] J. Quintero y R. Anaya. "Marco de Referencia para la evaluación de herramientas basadas en MDD". 10º Workshop Iberoamericano de Ingeniería De Requisitos y Ambientes Software IDEAS'2007, pp 225-238. 2007.

[32] A. Lins de Vasconcelos, G. Giachetti, B. Marín and O. Pastor. "Towards a CMMI-Compliant Goal-Oriented Software Process through Model-Driven Development. The Practice of Enterprise Modeling", pp. 253-267. ISBN: 978-3-642-24848-1. Publisher Springer Berlin Heidelberg. 201

ANEXO

CMMI-DEV en el desarrollo de software bajo el paradigma

Paradigma de Desarrollo Guiado por Modelos, MDD. En: Ventana Informática No. 35 (jul-dic).

Manizales (Colombia): Facultad de Ciencias e Ingeniería, Universidad de Manizales. p. 13-30.

ISSN: 0123-9678

CMMI-DEV en el desarrollo de software bajo el paradigma de Desarrollo Guiado por Modelos, MDD*[1]

[CMMI-DEV in software development under Model-Driven software Development, MDD]

[CMMI-DEV no desenvolvimento de software sob o paradigma de Desenvolvimento Model-Driven, MDD]

Rosalba MATOS[2], Claudia PONS[3]

Recibo: 10.02.2016 – Aprobación: 30.09.2016

Resumen: *Continuando con la tarea de buscar nuevos modelos de desarrollo que mejoren el proceso de construcción de software, surgió en Gran Bretaña en 1994, el concepto de Desarrollo Guiado por Modelos, MDD, con tres objetivos principales: Aumentar la portabilidad, la interoperabilidad y la reutilización del software. En la actualidad MDD está ganando atención de la industria y las comunidades de investigación. Ya se encuentran casos de éxito documentados que declaran un ahorro de tiempo significativo en proyectos de desarrollo bajo MDD, donde la Arquitectura dirigida por modelos (MDA), es uno de los modelos más conocido, desarrollada por Object Management Group (OMG). Este trabajo tiene como objetivo presentar para cada etapa del proceso de desarrollo bajo MDA, cuáles son los artefactos que deben tenerse en cuenta para*

* **Modelo para la citación de este artículo:**
MATOS, Rosalba & PONS, Claudia (2016). CMMI-DEV en el desarrollo de software bajo el paradigma de Desarrollo Guiado por Modelos, MDD. En: Ventana Informática No. 35 (jul-dic). Manizales (Colombia): Facultad de Ciencias e Ingeniería, Universidad de Manizales. p. 13-30. ISSN: 0123-9678

1 Artículo de investigación proveniente de la tesis *Cómo Integrar el Modelo CMMI al Modelo de Desarrollo de Software MDD*, para optar al título de Master en Ingeniería de software, por la Universidad Nacional de La Plata, durante el período *2013-2015*, por parte de la primera autora, bajo la dirección de la segunda.

2 Ing. de Sistemas; Esp. en Ingeniería de Software. Docente, Universidad Libre de Barranquilla (Barranquilla, Colombia). Correo electrónico: rosalba_matos@hotmail.com, rmatosm@unilibrebaq.edu.co

3 Licenciada en Informática, Especialista en docencia universitaria, Doctora en Ciencias Informáticas. Investigadora de la Comisión de Investigaciones Científicas CIC, Directora del Centro de Altos Estudios en Tecnología Informática (CAETI), Universidad Abierta Interamericana (Buenos Aires, Argentina). Correo electrónico: claudia.pons@uai.edu.ar

que un proyecto logre soportar 100% las prácticas específicas definidas por Capability Maturity Model Integration for Development (CMMI DEV 1.3), Niveles 2 y 3. Para ello se analizaron cada uno de los riesgos presentes en un desarrollo bajo MDD, y se establecieron recomendaciones que fueron validadas por varios investigadores sobre el tema de aplicación de los dos modelos CMMI y MDD en un mismo proyecto de desarrollo de software.

Palabras clave: *MDD, CMMI, MDA, OMG*

Abstract: *Continuing with the task of seeking new development models that improve the process of building software, the concept of Model-Driven Development, MDD emerged in Britain in 1994, with three main objectives: increasing portability, interoperability and reuse of software. Currently, the interest of industry and research communities on MDD is increasing so rapidly that it is already possible to find successful documented cases of successful declaring significant time savings in development projects under MDD. One of the best known MDD models is the MDA (Model Driven Architecture), developed by the Object Management Group - OMG. This work aims at presenting which artifacts are the most suitable to be considered in each stage of the development process under MDA, for a project to meet the specific practices defined by DEV CMMI 1.3, Levels 2 and 3. For this, each of the risks involved in a development under MDD were analyzed. Recommendations on the issue of implementation of both CMMI and MDD models in a software development project were defined and then validated with several researchers.*

Keywords*: MDD, CMMI, MDA, OMG.*

Resumo: *Continuando com a tarefa de buscar novos modelos de desenvolvimento que melhoram o processo de construção de software, surgiu na Grã-Bretanha em 1994, o conceito de Modelos de Desenvolvimento, MDD, com três objetivos principais: aumentar a portabilidade, interoperabilidade e reutilização software. MDD está ganhando a atenção das comunidades industriais e de investigação. Já documentados casos de sucesso de declarar uma significativa economia de tempo em projetos de desenvolvimento sob MDD, onde o Modelo Driven Architecture (MDA), é um dos modelos mais conhecidos, desenvolvido pelo Object Management Group (OMG). Este trabalho tem como objetivo apresentar para cada fase do processo de desenvolvimento sob MDA, os artefatos a ser tidos em conta são para um projeto para atingir 100% de apoio práticas específicas definidas por Capability Maturity Model*

Integration para o Desenvolvimento (CMMI DEV 1.3) os níveis 2 e 3. Foram analisados cada um dos riscos em desenvolvimento no âmbito MDD, e recomendações foram validados por diversos pesquisadores sobre o tema da aplicação de ambos os modelos CMMI e TDM no mesmo projeto de desenvolvimento estabelecido software.

Palavras-chave: *MDD, CMMI, MDA, OMG.*

Introducción

El avance tecnológico y el aumento de la complejidad de los sistemas han traído nuevos problemas para resolver a la Ingeniería de Software. Esto ha conducido a los investigadores a pensar en nuevas metodologías, herramientas y lenguajes que permitan realizar los procesos de la Ingeniería de Software de forma eficiente. El uso de modelos en el desarrollo de software surge como respuesta a esta necesidad, buscando lograr al final la generación automática del código, siendo el Desarrollo Guiado por Modelos (MDD), un ejemplo.

Debido a que la calidad del software depende de la calidad de los procesos de desarrollo, unir los beneficios de usar el paradigma de desarrollo dirigido por modelos a un modelo guía de mejora de los procesos de desarrollo, resulta muy prometedor. A pesar de esto, actualmente es poca la información que se encuentra sobre qué pautas tener en cuenta en un desarrollo MDD, para asegurar el éxito en las evaluaciones CMMI-DEV; aunque puede decirse que es posible encontrar mucha documentación de los modelos por separado.

Fueron varios los autores que contribuyeron con los resultados de sus investigaciones a reafirmar supuestos e identificar los artefactos que podrían aplicarse en un desarrollo MDD para cumplir con las guías CMMI-DEV. Entre ellos, Esterkin (2014, 100), quien al realizar el mapeo entre los dos modelos encontró que hay varias áreas de procesos y prácticas específicas de CMMI que no son soportadas por MDD.

Por otro lado, Quintero & Duitama (2014, 2) plantean los retos que aun afronta el MDD, como son las limitaciones de las herramientas y la falta de integración del modelo BPM en las transformaciones de modelos. Además, presentan las posibles razones para que puede fracasar un proyecto de desarrollo dirigido por modelos y proponen estrategias para mitigar ese riesgo, así como las características deseables en las herramientas, lenguajes y técnicas que soportan el desarrollo MDD.

Reforzando la justificación de incrementar y mejorar el uso de MDD, Martínez, Cachero & Meilá (2013, 189), concluyeron luego de realizar un experimento con 26 estudiantes de la Universidad de Alicante, que el paradigma de desarrollo dirigido por modelos es uno de los más difícil de entender, pero es el más útil en el largo plazo al compararlo con el modelo basado en código, y con el paradigma basado en modelos. De hecho, el experimento mostró cómo 20 de los 26 desarrolladores eligió MDD para continuar con el desarrollo del proyecto del experimento. Otro caso de estudio, es presentado por Teixeira & Dias (2015), con el objetivo de mostrar cómo se puede incrementar la productividad en el desarrollo de sistemas de software a través de la MDA, señalando la existencia de herramientas gratuitas que permiten desarrollar a bajo costo un sistema bajo el paradigma MDD.

El objetivo principal de este trabajo es brindar una guía al equipo de desarrollo de software bajo MDD, que le permita conocer qué artefactos debería implementar, si desea que sus procesos soporten las prácticas específicas definidas en CMMI DEV V.1.3, en los niveles 2 y 3.

El artículo está organizado de la siguiente forma: en la sección uno, se explican los principales conceptos teóricos necesarios para comprender la investigación, como son los procesos que se siguen en un desarrollo de software dirigido por modelos. En la segunda, se presenta la metodología utilizada en la investigación; en la sección tres se muestran y discuten los resultados obtenidos y, finalmente, las conclusiones y la bibliografía.

1. Fundamento teórico

MDD es un paradigma de desarrollo de software que utiliza modelos y cuyo objetivo es separar el diseño del sistema de la arquitectura de las tecnologías, para que puedan ser modificados independientemente. Busca tres objetivos principales: Portabilidad, la Interoperabilidad y la Reutilización que eventualmente deberían conducir a un aumento en la productividad. MDD se apoya en los modelos, las transformaciones entre modelos y los lenguajes específicos de dominio.

1.1 Proceso de desarrollo MDA (Arquitectura dirigida por modelos)

MDA surge aproximadamente en el año 2000 y, de acuerdo con Berre & Elvesæter (2008, 11), en el 2004 se acordó definirla durante una sesión plenaria del *Object and Reference Model Subcommittee*

of the Architecture Board (ORMSC), celebrada en Montreal: «*MDA es una iniciativa OMG que propone definir un conjunto de normas no patentadas que especificarán tecnologías interoperables con los que cuenta el desarrollo dirigido por modelos con transformaciones automatizadas*».

Es tanta la aceptación de MDA[4] que, según OMG (2015), apoya la evolución de las normas en dominios de aplicación tan diversos como la planificación de recursos empresariales, el control del tráfico aéreo y la investigación del genoma humano. Los procesos y actores principales de un proyecto de desarrollo (Tabla 1) bajo el paradigma MDA, definidas por Musat (2009, 20), son:

- El CIM: es el modelo más abstracto de todos los usados en MDA, y describe la lógica del dominio del negocio desde una perspectiva independiente de la computación.
- El PIM: es el segundo modelo en jerarquía de MDA. Describe todos los aspectos funcionales de la aplicación independientemente de la plataforma de software en la que se vaya a implementar y ejecutar.
- El PSM: es un modelo del sistema con detalles específicos de la plataforma en la que será implementado. Se genera a partir del PIM, así que representa el mismo sistema, pero con distinto nivel de abstracción.

Tabla 1. Actividades vs. Actores (Construido a partir de Mellor & Watson, 2004)

Actividad	ID del actor*						
	1	2	3	4	5	6	7
Identificar elementos del contexto del problema	X	X	X				
Seleccionar modelos		X	X	X			X
Afinar requerimientos			X	X			
Construcción y prueba del Modelo de Requerimientos	X		X	X			X
Transformar CIM en PIM y verificar completitud		X	X		X		X
Analizar la arquitectura			X	X			
Generar modelos PIM			X		X		X
Generar los modelos de diseño PSM´s		X	X	X	X		X
Generar el código básico	X	X	X			X	X

* Patrocinador, stakeholders, usuarios y clientes, (2) Líder del proyecto, (3) Arquitecto, (4) Analista de requerimientos, (5) Analistas diseñadores, (6) Analista / programadores, (7) Probadores y/o Auditores de Sistemas

4 Aborda el ciclo de vida completo de diseñar, desplegar, integrar y gestionar aplicaciones, así como el manejo de los datos usando estándares abiertos.

Las transformaciones para pasar de un modelo a otro, se constituyen en los pilares de MDA. Por ello, las herramientas de transformación de modelos juegan un papel importante en el desarrollo, y vale la pena revisar las características mínimas que se espera deben cumplir las herramientas de transformación de modelos para administrar los riesgos de las transformaciones.

Es recomendable que las herramientas de construcción y transformación de modelos cumplan las características de permitir a los desarrolladores ajustar las transformaciones y conocer el origen de cualquier elemento del modelo: trazabilidad, así como que los cambios hechos en el modelo inicial se conserven para cuando sea necesario volver a generar el modelo (consistencia incremental).

1.1.1 Generación del CIM. La figura 1 resume las actividades, secuencia y actores propios de la generación de un modelo independiente de la computación. Es pertinente considerar algunos de los riesgos que se deben administrar al elaborar un CIM:

- Que el modelo CIM tenga bajo nivel de exactitud.
- Que no contenga todas las reglas del negocio requeridas para el modelo del proceso de negocio que se modela.
- Según Quelopana, Vega & Meneses (2009, 5) es un riesgo que no se incluyan completamente todos los actores en el modelo, o sea, los primarios o tomadores de decisiones, los secundarios, o sea los que participan en la toma de decisiones, y los terciarios, que son los que producto del proceso de toma de decisiones, llevan a cabo las actividades o subprocesos operacionales.

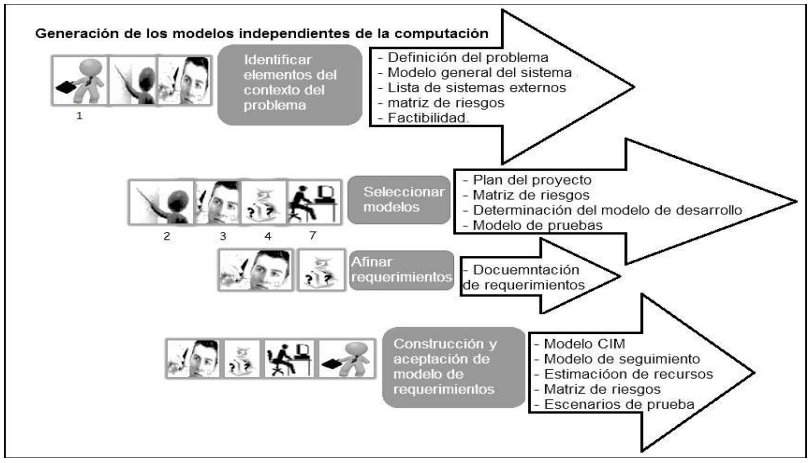


Figura 1. Actividades, y roles que intervienen en la generación del CIM

1.1.2 Generación del PIM. El PIM debe ser conductualmente completo (Figura 2), definiendo la lógica de negocio en términos de acciones abstractas. «*Lastimosamente, el proceso de desarrollo usando MDA, no considera la posibilidad de transformar el CIM en un PIM. Y aunque el CIM es bastante abstracto, parece necesario establecer un procedimiento que permita construir el PIM a partir del CIM, con la ayuda de una herramienta MDD y un conjunto de reglas de transformación*» (Chaparro & Gómez, 2012, 9).

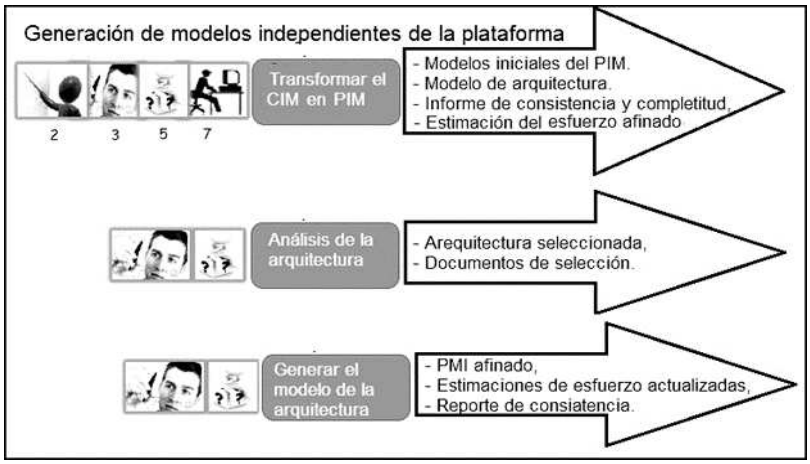


Figura 2. Actividades, y roles que intervienen en la generación del PIM

1.1.3 Generación del PSM. La figura 3a resume las actividades, secuencia y actores propios de la generación de un modelo específico de la Plataforma. Debido a que un PSM, es demasiado abstracto para la compilación en un idioma determinado, un entorno MDA requiere de un conjunto de reglas de transformación y una herramienta de Desarrollo de software guiado por modelos (MDD), para generar el código a partir del PSM. También puede ser necesario que se construyan varios PSM para los diferentes módulos del sistema, en cuyo caso también sería necesario utilizar Puentes, o herramientas de transformación, para que se comuniquen los diferentes PSM.

Los riesgos en esta etapa están inmersos en las siguientes actividades:

- Ejecutar reglas de transformación, - Verificar consistencia y completitud del modelo y - Verificar escenarios de prueba. Un riesgo que actualmente presentan las herramientas basadas en MDA es que no son fáciles de configurar inicialmente, ni de definir las transformaciones para una plataforma. Sin embargo, este esfuerzo es compensado cuando se utilizan estas transformaciones para múltiples aplicaciones.

Según Meaurio & Schmieder (2013, 145) «*MDA al permitir la doble transformación automática PIMPSM y PSM-código fuente hace que el esfuerzo se centre en los modelos de alto nivel, y al regenerar el resto de los modelos, la trazabilidad está garantizada. Esto hace que las iteraciones dentro del ciclo de vida sean más eficaces para afrontar estos cambios, minimizando de esta manera los riegos asociados*».

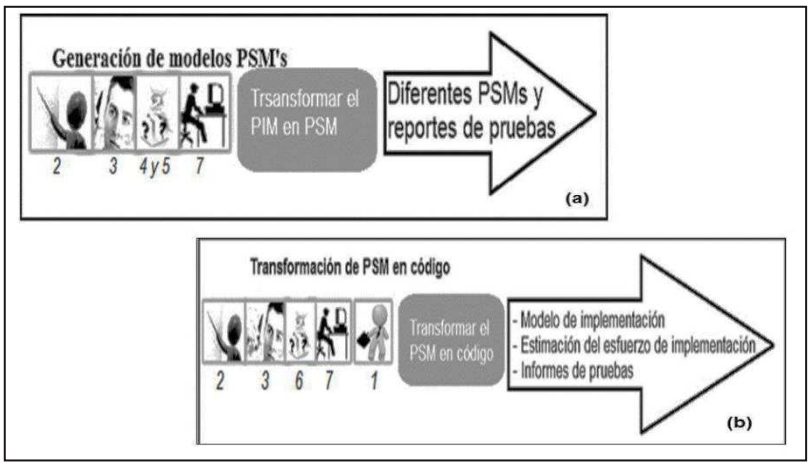


Figura 3. Actividades y roles

1.1.4 Generación del Código. La figura 3b resume las actividades, secuencia y actores propios de la generación del código. En esta actividad el Arquitecto, y el desarrollador utilizan las herramientas y reglas de transformación para generar el código básico de la aplicación que se desarrolla. Los Programadores al trabajar con herramientas MDA descubren que pueden escribir el código como parte de una plantilla de generación de código, dándoles más tiempo para trabajar en las piezas individuales más interesantes de la lógica que el código de un generador de código. La generación de código se puede hacer para varios lenguajes de programación como Java, C, Visual Basic.NET y C ++ entre otros.

1.2 CMMI (Capability Maturity Model Integration for Development)

CMMI-DEV «*proporciona una orientación para aplicar las buenas prácticas CMMI en una organización de desarrollo. Las buenas prácticas del modelo se centran en las actividades para desarrollar productos y servicios de calidad con el fin de cumplir las necesidades de clientes y usuarios finales*» (Chrissis, Konrad & Shrum, 2012). Con base en Puello (2013, 21-22), se hace un resumen histórico de CMMI:

- Sus principios se remontan a 1984, cuando en Estados Unidos se aprobó la creación de un organismo de investigación (SEI, Instituto de Ingeniería de Software) para la mejora en el desarrollo de los sistemas de software y evaluación de respuesta y fiabilidad de las compañías que suministran software al Departamento de Defensa.
- En 1985 el SEI origina el Modelo de Madurez de las Capacidades (CMM) y en 1991 publica la versión 1.0 del Modelo de Madurez de las Capacidades para el Software (SW-CMM, *Capability Maturity Model for Software*).
- A mediados de la década del 90, el SEI unifica los modelos de ingeniería de software (SW-CMM), de ingeniería de sistemas (SE-CMM) y de desarrollo integrado de productos (IPD-CMM), emergiendo una nueva generación llamada CMMISM (*Capability Maturity Model Integration*), que libera la versión 1.1 en 2002, que sirve de guía para mejorar los procesos organizacionales, además del desarrollo del Software.

1.3 Marco referencial

Algunos autores han abordado el tema de la utilización del modelo de madurez de capacidades CMMI-DEV en proyectos de desarrollo utilizando el paradigma de desarrollo dirigido por modelos:

- Esterkin (2014), presenta los resultados de un análisis para responder a la pregunta si las prácticas MDD, soportan el nivel de madurez 2 del CMMI. Como resultado del mapeo encuentra que hay varias áreas de procesos y prácticas específicas no soportadas por MDD.
- Esterkin & Pons (2012, 736-741), realizan un análisis de las buenas prácticas de MDD y determinan si éstas soportan las guías del nivel 2 de CMMI, y en qué medida las soportan.
- SEI (2010, 58), explica cómo interpretar las directrices del modelo CMMI para ser usadas en proyectos de desarrollo relacionadas con las Metodologías ágiles[5].
- En la búsqueda de una forma de evaluar la calidad en un proyecto de MDD, Ríos et al., (2006), muestran cómo un modelo dentro del proyecto *Modelware*[6] ayudó a varias compañías para adoptar el modelo de desarrollo MDD.

5 Esta posibilidad de usar el modelo de referencia CMMI y una metodología particular de desarrollo, ha servido de base para que Inigo Garro, consultor de CMMI, plantee cuáles son los aspectos claves que permiten la coexistencia de CMMI y los enfoques ágiles.

6 El proyecto *Modelware* fue desarrollado en España entre 2002 y 2006, definiendo cinco niveles de madurez que ubicaban a las empresas que adoptaban MDD en diferentes grados tanto de seguimiento de prácticas como de uso de artefactos.

2. Metodología

Esta investigación fue abordada desde un análisis de los riesgos de cada uno de los procesos en un desarrollo de software utilizando MDA, por ser uno de los modelos MDD más conocidos.

Luego de realizar las consultas pertinentes para conocer el estado del arte de ambos modelos, CMMI y MDD; confirmar la necesidad de vincular temas de mejora de procesos al paradigma de desarrollo MDD, y comprobar la escasez de trabajos documentados relacionados con el tema, se determinó que debía realizarse un análisis de los riesgos en cada una de las actividades de las etapas de desarrollo bajo MDA. Específicamente se analizaron los riesgos propios del modelo MDA, y luego se mapearon las actividades con riesgos, a las practicas específicas de las áreas de procesos CMMI-DEV 1.3

Por ser MDA un modelo de desarrollo, se limitaron las áreas de proceso que debían ser analizadas a las áreas de proceso que se centran en las prácticas específicas del desarrollo: Planificación del proyecto, Seguimiento y control del proyecto, Gestión de requerimientos, Gestión de la configuración, y Aseguramiento de la calidad del proceso y el producto, para el nivel 2, excluyendo Gestión de acuerdos con proveedores y Medición y análisis, que pueden llevarse de manera similar, independientemente de la metodología de desarrollo. Por la misma razón se incluyeron solo las áreas de Integración de Producto, Desarrollo de Requerimientos, Gestión de Riesgos, Solución Técnica, Integración del producto, Validación, Verificación, y Gestión integrada del proyecto para el nivel 3, y no se consideraron áreas para los niveles 4 y 5.

En la tarea de buscar los riesgos, se estableció como criterio de selección aquellos con alta probabilidad, debido a particularidades del MDD. Para ello, se analizaron para cada una de las etapas de desarrollo con MDA, el modelo correspondiente en cada etapa, las herramientas usadas, la transformación al siguiente modelo, y los riesgos existentes en cada transformación. Posteriormente se investigó sobre las actividades de cada etapa, las personas involucradas y por último de determinaron los controles que deberían implementarse para prevenir o minimizar los riesgos anteriormente identificados.

2.1 Validación de la propuesta

Con el fin de validar la propuesta, se realizó una encuesta[7] difundida en dos foros de CMMI, y un foro sobre MDD y se invitó por correo

7 La encuesta está disponible en: http://goo.gl/apcI50 , mientras sus resultados en goo.gl/vtSo2P

electrónico a algunos autores de artículos, libros y tesis citados en este trabajo, para que dieran su opinión sobre la propuesta. En busca de lograr la comprensión de la propuesta por parte de los colaboradores, se publicó un artículo previo, donde se presentó la propuesta.

3. Resultados y discusión

3.1 Descripción de resultados

Con base en los riesgos identificados, y tomando los resultados del análisis de Esterkin (2014, 80), sobre las prácticas específicas definidas por CMMI DEV 1.3 Nivel 2 soportadas en MDD, se describe el problema, y se identifican y proponen los artefactos que podrían utilizarse para satisfacer las prácticas CMMI que podrían no cumplirse. Además, se complementan las recomendaciones para soportar también CMMI DEV 1.3 Nivel 3. De esta forma se responde en parte la pregunta sobre como cumplir con CMMI-DEV 1.3 nivel 2 y 3, en un proyecto MDD. En la Tabla 2 se presenta el resumen de los resultados obtenidos en el proyecto[8].

Por otro lado, se encuentran algunos riesgos que se gestionan con la utilización de artefactos no propios de MDD, pero que deben ser especialmente atendidos por la misma naturaleza del desarrollo bajo MDD. Por ejemplo, es necesario establecer un marco de gestión de la configuración que ayude a controlar la transformación de modelos y facilite la verificación y la validación de la exactitud de los requerimientos. Por eso se propone utilizar los modelos de trazado a partir de metamodelos, o modelos de lenguajes de modelado, que facilitan la utilización de patrones dinámicos que estandarizan y guían la práctica de la trazabilidad a partir de los criterios de calidad. Es decir, ir más allá del uso de plantillas para formalizar las solicitudes de cambio. En cuanto a la matriz de riesgos, sería conveniente elaborarla teniendo en cuenta el registro de riesgos por actividades propias del desarrollo MDD.

8 Vale la pena aclarar que en esta tabla solo se mencionan las áreas de proceso y prácticas para las que se identificaron los riesgos altos de incumplimiento debido a las particularidades del desarrollo bajo MDD, que podrían ser mitigados con algún artefacto.

Tabla 2.

Resumen de los artefactos propuestos para cumplir con prácticas específicas de CMMI v1.3 niveles 2 y 3

Nivel	Área de proceso/ SP con alto riesgo propio en MDD	Problema	Artefacto propuesto
2	Gestión de requerimientos (REQM). - SP 1.4 Mantener una trazabilidad bidireccional de los requerimientos	«*Puede verse afectada en MDD por factores como la complejidad de los modelos, cambios invasivos no controlados, modelos parcialmente actualizados cuando hay cambios, matrices de trazabilidad no actualizadas, y altos costo para la realización de esta práctica*» (Tabares, 2009, 7).	Herramientas de gestión del trabajo y de configuración. MDD requiere de herramientas que van más allá de las matrices de trazabilidad[1]* de requerimientos. Utilizar los modelos de trazado a partir de metamodelos, o modelos de lenguajes de modelado, facilita la utilización de patrones dinámicos que estandarizan y guían la práctica de la trazabilidad a partir de los criterios de calidad.
	Seguimiento y Control de Proyecto (PMC). Se tienen cinco prácticas específicas no soportadas.	Que no se incluyan completamente todos los actores en el modelo. Cada actor realiza actividades críticas, puntuales y diferentes que difícilmente podrían ser ejecutadas por otro actor.	Especial cuidado en la asignación y control de participación de cada rol en las etapas del desarrollo (Tabla 1). Como las prácticas se pueden conducir en MDD, de manera análoga a un desarrollo tradicional, se analiza la práctica que merece tratarse diferente según SP 1.5 (Monitorear la Participación de los *stakeholders*)
	Planeamiento del Proyecto (PP). - SP 3.1 Revisar los planes que afectan el proyecto. - SP 3.2 Conciliar los niveles de trabajo y de recursos.	MDD no tiene prácticas propias que soporten las prácticas específicas 3.1 y 3.2, Además, en los desarrollos dirigidos por modelos es especialmente importante planificar el alcance, la estimación de recursos y la matriz de riesgos.	Artefactos usados en un desarrollo tradicional (registro de revisiones de planes que afectan el proyecto, negociación de recursos para conciliar lo planeado y disponible). El líder, el Arquitecto y el Analista de Requerimientos aseguran se incluyan las reglas del negocio requeridas, se verifique y valide por el probador y los usuarios, usan técnicas y tecnologías de soporte de la construcción sin ambigüedades[2]*.
	Gestión de la Configuración (CM). - SP 2.1 Rastrear los pedidos de cambio.	En el desarrollo dirigido por modelos es importante establecer un marco de gestión de la configuración que ayude a controlar la transformación de modelos y facilite la verificación y la validación de la exactitud de los requerimientos.	Se hace necesario fortalecer las prácticas de Gestión de la configuración y utilizar más que plantillas para formalizar las solicitudes de cambio. Utilizar los modelos de trazado a partir de metamodelos, o modelos de lenguajes de modelado, facilita la utilización de patrones dinámicos que estandarizan y guían la práctica de la trazabilidad[3]* a partir de los criterios de calidad.

	Aseguramiento de la Calidad del Proceso y del Producto (PPQA). - SP 2.1 Comunicar y solucionar problemas no resueltos.	«*Falta de procedimientos que indiquen la necesidad de registrar los problemas de incumplimiento*» (Esterkin, 2014, 97). La trazabilidad de requerimientos es un atributo de calidad tratado por la ingeniería de software, y el desarrollo por Modelos, requiere control de su consistencia y completitud.	Se deberían implementar reportes de acciones correctivas y reportes de evaluación. El modelo de trazado asumido en el proyecto de desarrollo y definido en el Plan de Calidad de la empresa son herramientas de Aseguramiento de la Calidad del producto y del proceso
3	Integración de Producto (PI). - SP 3.2 Ensamblar los componentes de producto. - SP 3.3 Evaluar los componentes de producto ensamblados.	El producto es construido a partir de componentes heterogéneos, ignorando las diferencias de implementación entre los componentes o aplicaciones. Así MDD atiende por naturaleza el asunto de la integración, sin descuidar el cumplimiento de estándares de los componentes.	Se recomienda tener un mecanismo para certificar que los artefactos y modelos cumplan los estándares y se mantenga la integridad del sistema
	Desarrollo de Requerimientos (RD). - SP 1.2 trasformar las necesidades de las partes interesadas en requisitos de cliente.	En un desarrollo MDD, cuando el modelado inicia en el nivel de requerimientos, se construye el Modelo Independiente de la Computación, CIM[4*], con las reglas de negocio y los requerimientos del sistema. Lo ideal es disponer de herramientas que permitan la transformación CIM a PIM, aunque se dificulte porque el CIM es bastante abstracto.	Se recomienda utilizar herramientas que faciliten la transformación automática de CIM a PIM. En los casos en que el equipo decida iniciar la construcción de modelos posterior a la etapa de refinamiento de requerimientos, el equipo deberá definir los artefactos que serán los ejes de trazado de requerimientos. «*Este procedimiento podría utilizar una herramienta MDD y un conjunto de reglas de transformación*» (Chaparro & Gómez, 2012, 77).
	Gestión de Riesgos (RSKM). - SP 1.3 establecer una estrategia de gestión de riesgos. - SP 2.1 Identificar los riesgos.	La gestión de riesgos es importante en MDD ya que cada construcción y transformación de modelos tiene riesgos que deben mitigarse en la construcción de un CIM y su transformación a PIM; Riesgos en la construcción de un PIM y su transformación a PSM; Riesgos en la construcción de un PSM y su transformación a código	Se recomienda elaborar la matriz de riesgos vs actividad de desarrollo. Esta matriz debería probarse y considerar los correctivos necesarios en las actividades de Construcción del modelo de requerimientos: Validaciones de completitud y consistencia, Transformación CIM a PIM, Generación de modelos de arquitectura, Transformación a código. En MDD, la transformación controlada por los modelos de trazado ayuda a disminuir los riesgos por subjetividad en las decisiones.
	Solución Técnica (TS). - SP 1.2 seleccionar las soluciones de componentes de producto.	Falta de definición de qué herramientas utilizar y qué modelos reutilizar.	Según Swithinbank et al. (2005, 66), el arquitecto de soluciones deberá elegir el tipo de modelo apropiado, (UML u otro), para que utilicen los desarrolladores de la aplicación. También deben definir las herramientas MDD necesarias para desarrollar el proyecto.
	Verificación (VER) - SP 3.1 realizar la verificación.	Según Esterkin & Pons (2012, 4), es necesario realizar verificación tanto del framework del modelo como de todos los artefactos generados.	Verificar en cada actividad de transformación de modelos entre lo construido y los requerimientos, y cada herramienta desarrollada. Los casos de prueba deben registrar automáticamente sus resultados.

1 Tabares (2009, 7) describe un nuevo enfoque de trazabilidad (Modelos de trazado) como un patrón que controla la transformación de modelos y atiende el problema de actualización con las matrices de trazabilidad, mejorando la consistencia y completitud de los modelos que se transforman.
2 Por ejemplo, para la construcción del CIM es recomendable el uso de un lenguaje gráfico notacional que permita la construcción de modelos más precisos que los construidos usando UML.
3 Definir modelos de trazabilidad como controladores de la transformación de los modelos en el contexto MDD mejora la gestión de la configuración en servicios como: la verificación de la consistencia y la completitud, y el manejo del cambio (Tabares, 2009, 78).
4 Existen varios estándares y herramientas que pueden utilizarse para la construcción del CIM y que proporcionan una serie de elementos para representar de manera estandarizada los métodos, ciclos de vida, roles, actividades, tareas y productos de trabajo utilizados en Ingeniería de software.

3.2 Discusión de resultados

Los resultados del trabajo fueron compartidos con las personas que respondieron una encuesta cuyo objetivo fue conocer la opinión de los expertos en ambos modelos, acerca de si las prácticas que se realizan en un proyecto de desarrollo de software que utilice el paradigma MDD son suficientes para cumplir 100% con las prácticas específicas definidas en CMMI DEV 1.3, en los niveles 2 y 3, o si es necesario la implementación de diferentes artefactos o enriquecer los que se aplican en un desarrollo tradicional, para su logro.

La encuesta fue respondida, por 73 personas[9], entre mayo 02 y septiembre 16 de 2015, con la única dificultad de conseguir personas conocedoras del paradigma de desarrollo MDD, que se sintieran seguros de dar su opinión. El 37% (25 personas) afirmaron conocer los dos modelos MDD y CMMI, de las cuales un 24% declararon que las guías actuales de CMMI no son suficientes para guiar los procesos en un desarrollo bajo MDD y el 44% que no lo saben; 79% señalaron su interés por conocer cómo usar ambos modelos en un mismo proyecto de desarrollo. Además, un 68% afirmaron que los modelos CMMI y MDD deberían complementarse en un proyecto de desarrollo MDD.

En relación con los artefactos identificados para ser implementados en un desarrollo dirigido por modelos con el fin de soportar las guías de calidad CMMI DEV nivel 2 y 3, no se obtuvo ninguna retroalimentación. Sin embargo, Pat O'Toole, un instructor certificado de CMMI-DEV (http://cmmiinstitute.com/conferences/speakers/pat-otoole), hizo un aporte importante relacionado con la propuesta inicial de incluir notas aclaratorias al modelo CMMI-DEV que facilitaran el cumplimiento, y la verificación de estos artefactos en un desarrollo MDD, de manera análoga a cómo se interpretan metodologías ágiles con CMMI. Su

9 11 desarrolladores de software, seis directores de proyectos, cinco SCAMPI Lead Appraiser, cuatro profesores de universidades, cuatro consultores, tres investigadores, y otros.

concepto fue que la incorporación de los conceptos y notas MDD en el CMMI sería un trabajo mucho más extenso, y dado el nuevo enfoque del SEI en el proyecto de próxima generación, como lo confirma Basu (2015, 37), es improbable que en este momento se realicen tales cambios. Atendiendo esta alarma, se cambió la idea de sugerir la incorporación de notas, por la recomendación de que los artefactos sean considerados en una Process Asset Library (PAL), que implemente un proceso estándar para la organización, siguiendo los estándares de CMMI.

4. Conclusiones

En este trabajo se abordaron dos modelos importantes en el desarrollo de software: Uno con mucho reconocimiento y muchos casos de éxito documentados en el ámbito mundial, autoridad en el tema de la mejora y evaluación de los procesos de desarrollo de software, y otro, que, a pesar de no estar maduro, promete darle un gran impulso a la industria del software, y cambiar el nivel de los procesos de desarrollo.

Al finalizar la investigación y luego de validar la propuesta, se pudo concluir lo siguiente:

- Es necesario documentar en un proyecto de desarrollo bajo MDD, la manera cómo se podría evaluar el cumplimiento de las guías CMMI-DEV.
- Existen riesgos que podrían afectar la calidad del producto, en un proyecto de desarrollo de software bajo MDD, que no son mitigados si se cumplen sólo prácticas del paradigma MDD o del desarrollo tradicional.
- Es válida la propuesta de utilizar artefactos adicionales que permitan asegurar el cumplimiento de CMMI DEV en los niveles 2 y 3.
- Una buena forma de integrar los artefactos que permitan el cumplimento de prácticas específicas definidas por CMMI DEV 1.3 Nivel 2 y 3, sería considerarlas en una Process Asset Library, PAL.

Se justifica la continuación de estudios similares ya que los equipos de desarrollo de software que aplican MDD, necesitan de un estándar que defina los lineamientos y buenas prácticas del proceso de desarrollo, teniendo en cuenta los riesgos y particularidades del paradigma de desarrollo por modelos, y los evaluadores CMMI, debe cubrir las necesidades del sector que será cada vez creciente de equipos de desarrollo y empresas que decidan incorporar nuevos paradigmas de desarrollo de software.

Sin embargo, para que esa integración se dé, es necesario que los investigadores,

los fabricantes de herramientas MDD, la academia y el instituto CMMI, filial de Carnegie Innovations y quien es ahora responsable de todo lo relacionado con el entrenamiento, certificaciones, evaluaciones y mejora de procesos del modelo CMMI reúnan esfuerzos para interpretar las prácticas CMMI dentro de un proceso de desarrollo MDD. Esto incluiría por lo menos las siguientes actividades:

- Incorporar las notas aclaratorias necesarias en el modelo CMMI-DEV.
- Definir las prácticas a evaluar en cada proceso.
- Documentar y difundir las mejores prácticas como parte del modelo CMMI.
- Capacitar y certificar a los evaluadores de procesos de una empresa que desarrolle bajo MDD.

5. Agradecimientos

Se hace reconocimiento a la valiosa colaboración brindada por cada una de las personas que respondieron la encuesta de validación de la propuesta. Se agradece en especial a Pat O'Toole, quien generosamente realizó la revisión del artículo previo y de las preguntas de la encuesta realizada.

Referencias bibliográficas

BASU, Anirban (2015). Ware quality assurance, testing and metrics. New Delhi (India): Star Print-O-Bind. 301 p. ISBN: 978-81-203-5068-7.

BERRE, Arne-Jørgen & ELVESÆTER, Brian (2008). Model Based System Development: Part I – MDE, Model Driven Engineering [online]. In: BERRE, Arne-Jørgen. Course INF5120 - Modelbased System development: Lecture Notes for Course. Oslo (Norway): University of Oslo. 51 p. <http://www.uio.no/studier/emner/matnat/ifi/INF5120/v08/undervisningsmateriale/INF5120-Part-I-MDE.pdf> [consult: 15/04/2016]

CHAPARRO LEMUS, Luis Oliverio & GÓMEZ ESTUPIÑÁN, Juan Federico (2012). Una visión del desarrollo de software utilizando modelos [en línea]. En: Gerencia Tecnológica Informática, Vol. 11, No. 29. (Ene-Abr). Bucaramanga (Colombia): Universidad Industrial de Santander. p. 69-82. ISSN: 1657-8236 <http://revistas.uis.edu.co/index.php/revistagti/article/view/2818/3060> [consulta: 20/01/2016]

CHRISSIS, Mary Beth; KONRAD, Mike & SHRUM, Sandy (2012). CMMI para Desarrollo: Guía para la integración de procesos y la mejora de productos. Madrid (España): Editorial Universitaria Ramón Areces. 722 p. ISBN: 978-8499610788

ESTERKIN, Viviana & PONS, Claudia (2012). Análisis y Evaluación del MDD (Model Driven software Development) desde la Perspectiva del Nivel 2 del CMMI -DEV 1.3 [en línea]. En: XVIII Congreso Argentino de Ciencias de la Computación, CACID 2012 (08 -12/10/2012), Bahía Blanca (Buenos Aires, Argentina): Red de Universidades con Carreras en Informática, RedUNCI. Anales CACID 2012, p. 734-743. <http://sedici.unlp.edu.ar/handle/10915/23705>, <http://sedici.unlp.edu.ar/bitstream/handle/10915/23705/Documento_completo.pdf?sequence=1> [consulta: 14/03/2015]

ESTERKIN, Viviana (2014). Análisis y Evaluación del MDD (Model Driven Development) desde la perspectiva de CMMI DEV 1.3 Nivel 2. Tesis de grado (Magister en Tecnología Informática). Buenos Aires (Argentina): Universidad Abierta Interamericana, UAI. 161 p.

MARTÍNEZ ESPINOSA, Yulkeidi; CACHERO, Cristina & MELIÁ, Santiago (2013). MDD vs. traditional software development: A practitioner's subjective perspective [online]. In: Information and Software Technology, Vol. 55, No. 2 (feb). Newton (MA, USA): Butterworth-Heinemann. p. 189-200. ISSN: 0950-5849. <http://dx.doi.org/10.1016/j.infsof.2012.07.004>, <http://www.sciencedirect.com/science/article/pii/S0950584912001309> [consult: 12/09/2016]

MEAURIO, Valeria S. & SCHMIEDER, Eric (2013). La Arquitectura de Software en el Proceso de Desarrollo [en línea]. En: Revista Latinoamericana de Ingeniería de Software, Vol. 1, No. 4, Lanús (Buenos Aires, Argentina): Red de Ingeniería de Software de Latinoamérica, RedISLA. p. 142-146. ISSN: 2314-2642 <http://revistas.unla.edu.ar/software/article/download/103/77> [consulta: 10/02/2016].

MELLOR, Stephen J. & Watson Andrew (2004). Roles in the MDA Process [on line]. Alabama (USA): Object Management Group, OMG. <http://www.omg.org/registration/Roles_in_MDA1.pdf> [consult: 17/07/2016].

MUSAT SALVADOR, David (2009). Espora: Definición de lenguajes de operación específicos de dominios siguiendo un proceso de desarrollo dirigido por modelos [en línea]. Proyecto fin de carrera (Ingeniero Técnico en Informática de Sistemas). Madrid (España): Universidad Politécnica de Madrid, Escuela Universitaria de Informática, Departamento de Organización y Estructura de la Información. 102 p. <http://oa.upm.es/1392/2/PFC_DAVD_MUSAT_SALVADOR_DEFINITIVO.pdf> [consulta: 17/07/2016]

OBJECT MANAGEMENT GROUP, OMG (2015): Model Driven Architecture: The Architecture of Choice for a Changing World - Executive Overview [online]. Needham (MA, USA): OMG. (Last updated: 31/05/2015). <http://www.omg.org/mda/executive_overview.htm> [consult: 25/04/2015].

PUELLO, Oswaldo R. (2013). Modelo de verificación y validación basado en CMMI [en línea]. En: Investigación e Innovación en Ingenierías, Vol. 1, No. 1 (ene-jun). Barranquilla (Atlántico, Colombia): Universidad Simón Bolívar, Instituto de Investigaciones Científicas. p. 20-27. ISSN: 2344-8652 <http://publicaciones.unisimonbolivar.edu.co:82/rdigital/ojs/index.php/innovacioning/article/view/478/474> [consulta: 16/07/2016]

QUELOPANA, Aldo; VEGA Z., Vianca & MENESES V., Claudio (2009). Una propuesta metodológica para modelar procesos de negocio de decisión basada en una extensión a BPMN [en línea]. En: 3er Encuentro Informática y Gestión: Workshop Internacional EIG2009 (03-04/12/2009). Temuco (Chile): Universidad de La Frontera. CEUR Workshop Proceedings. ISBN: 978-956-236-204-7 <http://ceur-ws.org/Vol-558/Art_12.pdf> [consulta: 13/03/2016]

QUINTERO, Juan Bernardo & DUITAMA MUÑOZ, Jhon Freddy (2014). Reflexiones acerca de la adopción de enfoques centrados en modelos en el desarrollo de software [en línea]. En: Ingeniería y Universidad, Vol. 15, No. 1 (ene-jun). Bogotá (Colombia): Pontificia Universidad Javeriana. p. 219-243. ISSN: 0123-2126. <http://revistas.javeriana.edu.co/index.php/iyu/article/view/1131/810> [consulta: 12/02/2016]

RIOS, Erkuden; BOZHEVA, Teodora; BEDIAGA, Aitor & GUILLOREAU, Nathalie (2006). MDD Maturity Model: A Roadmap for Introducing Model-Driven Development. In: Second European Conference on Model Driven Architecture® Foundations and Applications, ECMDA-FA'06 (10-13/07/2006), Bilbao (Spain): ECMDA. RENSIK, Arend & WARMER, Jos (eds.). Model Driven Architecture – Foundations and Applications: Proceedings of the ECMDA-FA'06, Berlin (Germany): Springer. p. 78-89. ISBN: 3-540-35909-5

SOFWARE ENGINEERING INSTITUTE, SEI (2010). CMMI para desarrollo, Versión 1.3: CMMI-DEV, V1.3. Pittsburgh (USA): Carnegie Mellon University. 482 p. ISBN: 978-1-4467-5714-7.

SWITHINBANK, Peter; CHESSELL, Mandy; GARDNER, Tracy; GRIFFIN, Catherine; MAN, Jessica; WYLIE, Helen & YUSUF, Larry (2005). Patterns: Model-Driven Development Using IBM Rational Software Architect [Redbooks]. (USA). IBM, International Technical Support Organization. 236 p. ISBN: 9780738492889. <http://www.redbooks.ibm.com/redbooks/pdfs/sg247105.pdf> [consult: 13/10/2015]

TABARES BETANCUR, Marta Silvia (2009). Un patrón de trazabilidad para controlar la evolución de los intereses en un espacio multidimensional. Tesis doctoral (Doctor en Filosofía de Ingeniería -Sistemas) Medellín (Colombia): Universidad Nacional de Colombia, Escuela de Sistemas e Informática. 272 p.

TEIXEIRA COSTA, Pedro Henrique & DIAS CANEDO, Edna (2015). Using a MDD approach to develop software systems [online]. In: 10th Iberian Conference on Information Systems and Technologies, CISTI 2015 (17-20/06/2015). Aveiro (Portugal): AISTI, Associação Ibérica de Sistemas e Tecnologias de Informação. Proceedings of the CISTI 2015. Washington, D.C. (USA): Institute of Electrical and Electronics Engineers, IEEE. p. 108-113. e-ISBN: 978-9-8998-4345-5. <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7170371>, <DOI: 10.1109/CISTI.2015.7170371> [consult: 01/10/2016]