Hacia un Modelo de Madurez para apoyar el Desarrollo de Software Dirigido por Modelos

Valenzuela, Jorge., Pavlich-Mariscal, Jaime A.
{jorge.valenzuela, pavlich} @javeriana.edu.co
Pontificia Universidad javeriana

***Abstract*— Global software market is more demanding. Model driven development (MDD), which builds software from models and not from code, promises quality and productivity improvement. A software process model supporting the MDD paradigm would hopefully facilitate the adoption of MDD. This paper proposes a software process improvement model oriented to small organizations, including goals and best practices, based on standards, and complemented by specific elements for MDD. The model also includes an assessment method aimed to determine the current state of the organization and to indicate improvement plans to gradually evolve across the model. Model and method are being applied in or order to evaluate its benefits.**

Keywords—Software Engineering, Model-Driven Development, Capability Maturity Models

I. INTRODUCCIÓN

Se hablaba de crisis del software antes que la estructuración formal de principios científicos promoviera la ingeniería para conjurar la problemática que tenían los grandes proyectos: atrasos, sobrecostos, errores, ineficiencia [1]. Aún hoy los problemas [2] siguen vigentes dentro de una industria que se ha consolidado mayoritariamente por pequeñas y medianas organizaciones [3].

El escenario actual muestra una demanda creciente de servicios y productos de software [4] con mayor exigencia en calidad y plazos de entrega frente a una oferta conformada en su mayoría por pequeñas y medianas organizaciones de desarrollo de software. Un mercado global significa nuevas oportunidades pero también mayores exigencias [5]; la competitividad y la supervivencia de las pequeñas organizaciones de desarrollo de software dependen de incrementar calidad y reducir plazos de entrega de productos y servicios.

Práctica e investigación de la ingeniería de software en mejora de procesos [6] dieron origen a modelos de madurez y capacidad, dirigidos a incrementar la calidad y la productividad del desarrollo de software.

Idénticos objetivos se buscan planteando cambios de paradigma [7], que proponen obtener el software desde un mayor nivel de abstracción, con modelos que permitan mejor expresión y razonamiento sobre el dominio y el problema a resolver, en vez de producirlo escribiendo código. Esto se denomina Desarrollo Dirigido por Modelos (MDD) [8].

Conjugar ambas promesas aumentaría las probabilidades de incrementar calidad y productividad en la industria del software. Es la tesis que desarrolla este artículo y que se ilustra mejor con la **Fig. 1** que clasifica aproximaciones existentes dependiendo del tamaño de la organización a la cual van dirigidos (eje horizontal) y la especificidad del paradigma de desarrollo utilizado (convencional, vs MDD):

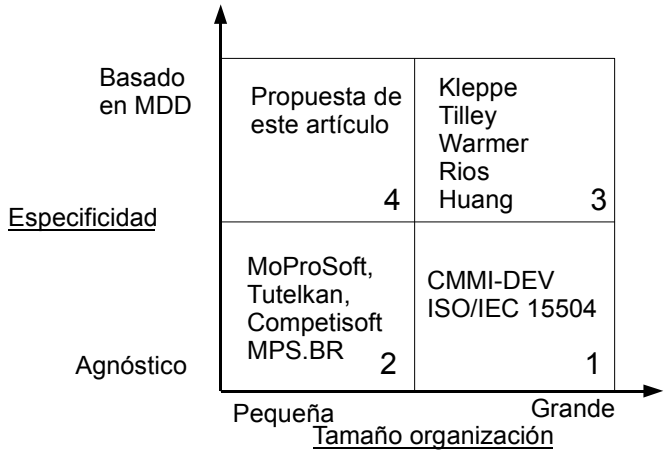

Fig. 1. Clasificación de modelos de madurez de acuerdo a complejidad y especificidad.

El cuadrante 1 incluye modelos como CMMI-DEV [9] e ISO/IEC 15504 [10], aproximaciones para grandes organizaciones y con alto grado de desconocimiento respecto al paradigma de desarrollo. El cuadrante 2 incluye propuestas regionales como MoProSoft [11], Competisoft [12], MPS.BR. [13], Tutelkan Reference Process [14] o Agile-SPI [15], que se orientan a pequeñas y medianas empresas (MiPyMES) de software en Latinoamérica, quienes conforman la gran mayoría del sector [16], propuestas de menor complejidad, pero que no hacen explícito el uso de modelos para dirigir el desarrollo.

El cuadrante 3 incluye modelos de madurez y capacidad que utilizan explícitamente MDD [17] [18] [19] [20] [21] [22], [23]. Estas propuestas no están adaptadas para pequeñas y medianas organizaciones.

El foco de la presente investigación es la propuesta de modelo de madurez y capacidad dentro del cuadrante 4 de la figura: Un modelo ágil y específico para MDD, aplicable en pequeñas organizaciones de desarrollo de software.

978-1-4799-6717-9/14/$31.00 ©2014 IEEE

II. CONTEXTO CONCEPTUAL

A. Desarrollo Dirigido Por Modelos

Los modelos constituyen un lenguaje de discusión y entendimiento [24], modelar es el camino entre comprender y hacer, utilidad práctica para la resolución de problemas [25] y apoyo al desarrollo de software. Las primeras propuestas emergieron hace más de 3 décadas: el modelado conceptual de datos [26] y las herramientas CASE [27].

El tema resurge con diversas propuestas y alternativas: Model-integrated Computing (MIC) [28], Unified Modeling Language (UML) [29], Model Driven Architecture (MDA) [30], Software Factories (SF), que se resumen en el Desarrollo Dirigido por Modelos (MDD).

Un modelo trasciende su representación gráfica (diagrama), es un constructo lingüístico con sintaxis y semántica (notación y significado) y cuyas declaraciones (proposiciones con valor de verdad) o bien describen un sistema bajo estudio, o bien prescriben (o especifican) un sistema posible, [31]. El fin de MDD es obtener el sistema directamente desde los modelos, pasar del modelo descriptivo al modelo prescriptivo, hacer que el modelo sea el sistema.

Un modelo descriptivo es válido si todas sus proposiciones son verdaderas, un modelo prescriptivo se considera válido por principio y el sistema que se construya (y que represente) debe tener conformidad con sus proposiciones [31].

Además de la validez del modelo en relación con la realidad que representa (descriptiva o prescriptiva), los modelos deben tener conformidad con la estructura lógica del lenguaje que determina las reglas de las expresiones bien formadas con los símbolos de su notación. Esto se denomina la sintaxis abstracta del lenguaje de modelado.

Si la sintaxis abstracta y la sintaxis concreta delimitan el dominio semántico de los modelos válidos (restringe los conceptos de una realidad o sistema particular), tenemos un lenguaje de modelado de dominio específico, en caso contrario tendremos un lenguaje de modelado de propósito general..

B. Modelos de Madurez

Los modelos de madurez y capacidad responden a las preocupaciones de investigación y práctica de la ingeniería de software denominado mejora de procesos (Software Process Improvement – SPI), con principios ligados a la preocupación devenida de los problemas que tenían los proyectos. Los primeros textos [32] pretendían agregar formalidad y estructuración a las actividades relacionadas al desarrollo de software, su propuesta establecía 4 etapas, la última se denominaba madurez.

La premisas fundamentales se tomaron del movimiento sobre el manejo total de la calidad [33], del trabajo sobre control estadístico [34] y de la matriz de madurez de la calidad [35], que se consolida en un primer modelo de madurez y capacidad en [6], con mayor formalismo y complejidad.

Los objetivos fundamentales de SPI son el aumento de la calidad y la reducción de costos y tiempos en la producción de software [36]. Los estándares internacionales de SPI más conocidos son los modelos de madurez y capacidad [9], [10]. Se componen de una taxonomía de procesos de referencia y un método de valoración que se construye sobre el concepto de niveles y su evolución, como métrica para evaluar y calificar los procesos, a través de metas definidas que para cumplirse requieren seguir un conjunto de buenas prácticas.

1*) Críticas y reacciones*: Los fundamentos de estos modelos muestran influencia de áreas diferentes a la ingeniería de software, dando argumentos a [37] para replantear sus bases ontológicas y epistemológicas enfocándolas hacía cuerpos de conocimiento más cercanos o apropiados a la tecnología, como la Ciencia del Diseño. Otra opinión juzga la vaguedad de estos modelos, dada su generalidad descriptiva y su abstracción prescriptiva, afirma que hay una brecha entre saber y hacer que a veces no produce la suficiente acción [38]. Críticas adicionales se hallan en [39], sobre la reverencia que se hace a los procesos ignorando las personas. Herbsleb et al [40], reprueban la burocracia que produce la adopción y cumplimiento de los modelos y previenen sobre el efecto posible en las personas, cortando la innovación. Otra advertencia la provee [36] que recuerda que la mejora de procesos es un medio para un fin: mejores resultados para el negocio. Una objeción importante a los estándares internacionales proviene de su tamaño y complejidad frente a la realidad de la mayoría de empresas de software en el mundo que son pequeñas organizaciones a las cuales les resulta arduo el esfuerzo financiero y humano para lograr conformidad [3]. Sin embargo se plantea como útil y deseable que estas empresas obtengan los beneficios de aplicar un modelo de madurez para procesos de software.

Los países iberoamericanos no desconocieron las críticas y emergieron propuestas para MiPyMES en contextos regionales o nacionales [11], [12], [13], [14] y [15].

Las organizaciones de estándares reaccionaron, la ISO/IEC publicó un conjunto de normas y reportes técnicos desarrollados de acuerdo a las características y necesidades de pequeñas y medianas organizaciones de software de no más de 25 personas: las normas ISO/IEC 29110 para pequeñas organizaciones [41]. El SEI hizo precisiones y consideraciones especiales para pequeñas empresas [42] en entornos ágiles.

2) El movimiento ágil, la mejora de procesos y MDD: Otra reacción notable fue el movimiento de desarrollo ágil. Según Fowler [43], la excesiva ceremonia y exigencia de las alternativas basadas en procesos formales, fue argumento fundamental para el nacimiento de propuestas que privilegiaban otros valores [44].

Una mirada a propósitos y objetivos permite coincidir y reforzar en lo común y distinguir y complementar en lo diverso: SPI, por una parte, pretende una representación explicita de conocimiento válido y relevante dirigido a conformar capital intelectual organizacional, conocimiento que trasciende las actividades del desarrollo de software. Por otra parte, el fin primordial de los movimientos agiles se resume en su manifiesto y sus principios, casi exclusivamente dirigidos al desarrollo de software. Considerando propósito y alcance, no son comparables, la Agilidad maneja un conocimiento tácito basado en las personas y SPI un conocimiento explícito para la creación continua de conocimiento como activo de una organización. Por otra parte,

978-1-4799-6717-9/14/$31.00 ©2014 IEEE

SPI describe el "qué" se debe hacer o conseguir y la agilidad prescribe el "cómo" se debe actuar para lograr ese "qué". Resumiendo, SPI define estrategia y Agilidad determina táctica; SPI proporciona modelos descriptivos, la Agilidad prescribe metodologías.

A pesar de las diferencias su asociación es simbiótica [45], entendiendo que Agilidad no significa exactamente liviano o anárquico sino flexible y con entregas frecuentes y rápidas de software pero sin olvidar la disciplina. [46].

Conciliar agilidad con MDD también es tarea ardua y con oposición desde los extremos; quienes piensan que un modelo es documentación con escaso o nulo valor en el desarrollo de software y quienes consideran que en MDD no hay lugar al código.

III. DESCRIPCIÓN DE LA PROPUESTA

El modelo propuesto parte de los grandes estándares y se conjuga en dos partes principales: 1) Un modelo de procesos de referencia con niveles incrementales de madurez y capacidad; 2) Un método de evaluación y diagnóstico.

La estrategia de adaptación hacia pequeñas organizaciones es aligerar el modelo. Se reduce el número de elementos dentro de los dos modelos base y se adecuan y complementan los elementos seleccionados, orientándolos hacia una forma más ágil y pertinente con MDD. La estructura fundamental del modelo de procesos se toma de [9] junto con los procesos de ISO/IEC 12207 [47] (modelo de procesos de [10]). La selección deja diez áreas de proceso a las que se añaden tres de una categoría específica para MDD. Es de notar que ninguna de las áreas consideradas en el modelo supera el nivel 3 de madurez, lo que descarta los niveles 4 y 5. Estos niveles son los que contienen las áreas denominadas de alta madurez caracterizadas por un mejoramiento cuantitativo que demanda mayores recursos y esfuerzos que los niveles inferiores para lograr su implementación y conformidad. Las estadísticas muestran que más del 90% de las organizaciones evaluadas por el SEI están en un nivel 3 o inferior y que adquiriendo estos niveles han obtenido ventajas significativas. Las organizaciones que buscan los niveles de alta madurez, 4 y 5, han trascendido de tal forma que posiblemente ya no son pequeñas y muy pequeñas empresas de software sino industrias de mayor tamaño con más recursos para emprender un programa de mejoramiento de procesos. El modelo propuesto está orientado a pequeñas y muy pequeñas organizaciones y por tal razón se delimita el máximo nivel a cuatro y se renombran los niveles así: 1 – Basado en Modelos; 2 – Soportado en Modelos; 3 – Dirigido por modelos y 4 – Optimizado por modelos, se agrega un nivel inferior de capacidad, 0 - inmaduro.

La **Fig. 2**, muestra las relaciones de las áreas de proceso para entornos ágiles ubicadas en su respectivo nivel y agrupadas por categoría, conservando la nomenclatura y significado que les da [9]; se complementa con la categoría MDD y sus áreas de proceso. En el nivel 2 se tienen dos áreas: modelado de requerimientos (RMOD) que representa el espacio del problema y sus requerimientos y Modelado de Diseño (DMOD) que especifica el espacio de la solución en el diseño y la arquitectura. En el nivel 3 se tienen dos áreas: transformación parcial (TRA) y transformación total o generación de código o modelos ejecutables o interpretables (GEN). En el nivel 4, se tiene un área de procesos (MDSE) cuyo propósito es el uso sistemático de los modelos como principal artefacto en el ciclo de vida del software con la utilización de modelos formales.

Como se puede apreciar, las áreas de MDD se relacionan en doble sentido con la categoría de Procesos y de Ingeniería y le aportan información a las categorías de Soporte y de Proyectos.

El resultado es un conjunto de 33 metas específicas que se agrupan por categoría: Proyectos 9, Ingeniería 12, Soporte 5 y MDD 7. La categoría de Procesos se incluye tácitamente con las metas y prácticas genéricas de [9], que determinan la

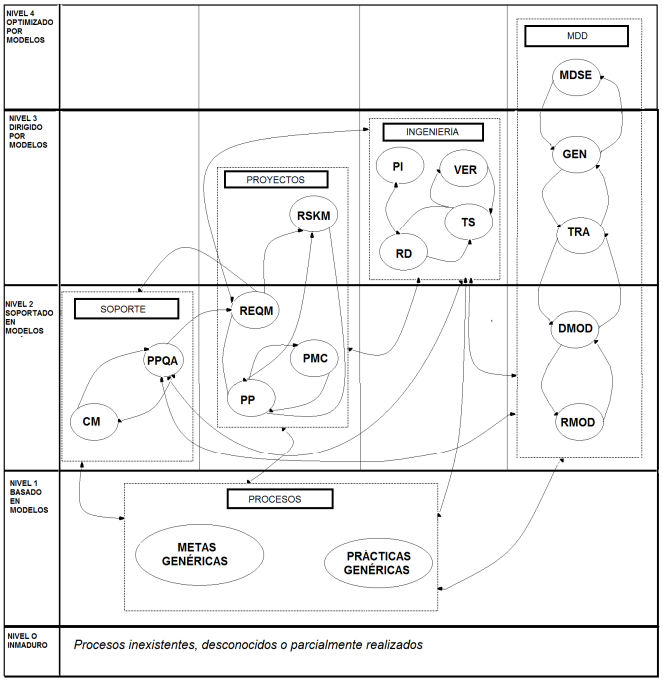

Fig. 2. Relación entre áreas y categorías del modelo

realización, la gestión y la definición de un proceso y se hacen efectivas siguiendo el estándar ISO/IEC 29110 [41].

Los componentes del modelo de procesos y sus relaciones, se presentan en el diagrama conceptual de la figura **Fig. 3**, las categorías agrupan las áreas de proceso, que a su vez hacen parte de un nivel de madurez o capacidad y que se satisfacen o cumplen a través de metas que se logran con la implementación de ciertas prácticas, específicas o genéricas. La evolución puede tomar dos caminos o representaciones, la escalonada que representa la madurez y toma un conjunto predefinido de áreas de proceso y la continua, asociada a la capacidad y dirigida a mejorar áreas individuales de proceso.

I. VALIDACIÓN DEL MODELO

Para el componente de evaluación y diagnóstico se tomó como base el método estándar SCAMPI. La reducción y adaptación a pequeñas organizaciones se consiguió aplicando una versión ligera del método que utiliza dos instrumentos: una

978-1-4799-6717-9/14/$31.00 ©2014 IEEE

encuesta de cinco cuestionarios de opción múltiple y cerrada de tipo Likert y una plantilla de verificación de la existencia de artefactos o productos de trabajo.

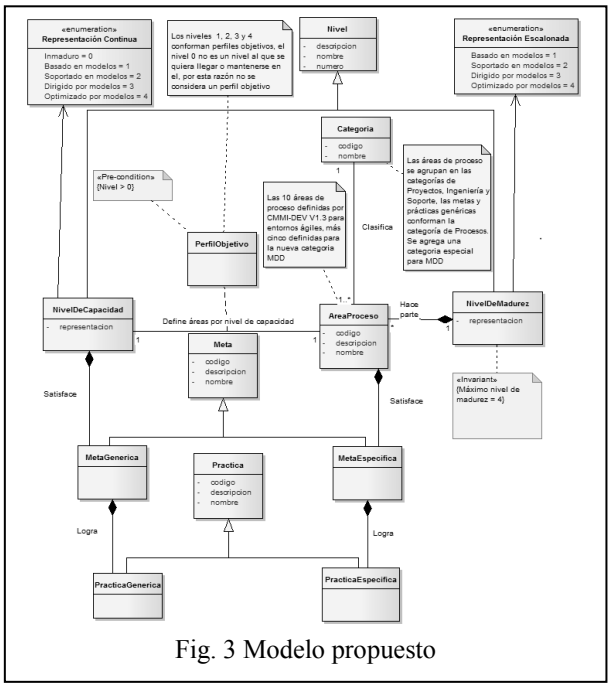

Fig. 3 Modelo propuesto

Los instrumentos se aplicaron a tres grupos de trabajo en una entidad del sector público cuya área de informática responde continuamente a requerimientos de tecnología. Los tres grupos suman diecisiete personas que se ocupan en actividades de desarrollo y/o mantenimiento de software con asignación exclusiva a sendos proyectos, estos proyectos cuentan con tres, tres y once personas respectivamente

El primer instrumento obtuvo una autovaloración de las personas que participan en los proyectos, para recabar las afirmaciones que constituyen parte de la evidencia requerida; el segundo instrumento corroboró o desmintió el cumplimiento de las metas con la aplicación de las prácticas del modelo, a través de la verificación de artefactos o productos.

Para el reconocimiento inicial de la organización se hizo una valoración a través de una entrevista al nivel directivo que estableció que la organización cuenta con algunos lineamientos y estándares metodológicos para las actividades de desarrollo de software.

Para elaborar las preguntas de la encuesta, esta se dividió en cinco cuestionarios, uno por cada categoría de áreas de proceso del modelo propuesto, para las categorías de Procesos y Proyectos se tomó como base los cuestionarios de Competisoft [48], para los otros cuestionarios se tomaron las prácticas (genéricas y específicas) del modelo. Cada cuestionario está compuesto por cinco ítems o reactivos que son las declaraciones en forma de pregunta que describen características de buenas prácticas.

El grado de ausencia o presencia de esas características se miden con una escala Likert [49] de cinco puntos, con cada opción asociada a un peso o ponderación ordinal (nunca = 1; Rara vez = 2; En ocasiones = 3; Con frecuencia = 4 y Siempre =5).

La variable evaluada es el nivel de la organización frente al modelo propuesto de acuerdo con los criterios de una evaluación SCAMPI, en particular con la tabla que caracteriza la implementación de las prácticas del modelo, de donde se toma las caracterizaciones y se hace una correspondencia con los valores de respuesta de la escala Likert de los cuestionarios, como se muestra en la tabla

Tabla 1. Correspondencia caracterización SCAMPI con escala Likert

Caracterización SCAMPI de la práctica	***Correspondencia en escala Likert***
Totalmente implementada (FI)	Siempre (5)
Ampliamente Implementada (LI)	Con frecuencia (4)
Parcialmente implementada (PI)	En ocasiones (3)
No implementada (NI)	Rara vez (2)
No todavía implementada (NY)	Nunca (1)

A partir de ésta correspondencia se usan las reglas de agregación SCAMPI para determinar una caracterización final de toda la categoría evaluada en cada cuestionario, tomando las caracterizaciones (frecuencias absolutas de respuestas) del cuestionario.

Los cuestionarios se respondieron de forma individual de acuerdo a la experiencia y participación dentro de la organización y contexto del desarrollo de Software.

La fiabilidad de cada cuestionario de la encuesta, se estimó, una vez respondido y tabulado, con el método de consistencia interna basado en el coeficiente Alfa (α) de Cronbach [50]. Este coeficiente se calculó conjugando algunas medidas estadísticas que permiten evaluar la confiabilidad o consistencia interna de instrumento constituido por una escala de opciones múltiples, tal como la escala Likert. Su ecuación es la siguiente:

$$\alpha = \frac{k}{k-1}\left(1 - \frac{\sum_{i=1}^{k} var(x_i)}{var(x)}\right)$$

En donde k representa la cantidad de ítems o preguntas del cuestionario y var(x) es la varianza de los puntajes obtenidos. El valor obtenido demostró una consistencia interna suficiente de los ítems analizados.

Sobre los datos tabulados de cada cuestionario, se hizo un análisis de estadística descriptiva con indicadores de tendencia central de moda, mediana, rango, rango inter-cuartílico y promedio para cada ítem. El rango para conocer la amplitud entre puntajes extremos y el rango inter cuartilico para conocer esa amplitud en el 50% de los datos, es decir desconociendo los valores extremos.

La plantilla de validación tomó el trabajo efectivo que se hizo en cada proyecto y lo contrastó contra los componentes del modelo de procesos para evaluarlo frente a una caracterización dada y así produjo una valoración, soportada en evidencia objetiva, que valoró el grado de satisfacción, de

cumplimiento o de conformidad contra lo descrito en las prácticas, metas o niveles del modelo.

Junto con la estimación del nivel, la valoración describió hallazgos en forma de debilidades y fortalezas que indicaron planes de mejora para reforzar las áreas y metas que mostraron debilidades en la evaluación. Los planes se prepararon para ser ejecutados en un periodo de dos meses con seguimiento mensual de avance.

Los resultados obtenidos al evaluar nuevamente los grupos de trabajo luego del vencimiento de la fecha final del plan, tan solo mostraron pequeños avances frente a la valoración inicial, principalmente porque muchas de las acciones del plan no fueron ejecutadas. Esto sucedió, en gran parte, porque los grupos de trabajo no contaron con el tiempo para implementar las prácticas del modelo. Se evidenció un bajo soporte de la dirección para impulsar la ejecución de los planes.

IV. TRABAJOS RELACIONADOS

Los modelos de madurez para MDD es tema explorado en trabajos previos: [17], [18], [20], [22], [21] [23]. En [17], el modelo está fuertemente orientado a la iniciativa MDA y a los estándares OMG para implementar una versión ejecutable de OCL. Presenta un modelo de niveles que pueden compararse a los niveles del CMM e indican qué rol juegan los modelos en el proceso de desarrollo de software y la dirección para mejorar este proceso.

En [18], se describe un modelo basado en niveles y dirigido a ayudar al entendimiento de los programas durante el desarrollo y en las actividades posteriores de mantenimiento o extensibilidad, el enfoque se centra en los aspectos de calidad de la documentación dando relevancia a UML como alternativa para lograr esto. El trabajo de [19] es un estudio del año 2008, da un vistazo a la adopción global de MDD y examina el valor que este enfoque ofrece a las Empresas de Software; concluye que MDD es una innovación que perdurará. Reconoce la amplia adopción y despliegue que está teniendo, además de los beneficios que trae a medida que madura su uso en las organizaciones. Propone un marco de madurez que consiste en tres niveles, cada nivel se caracteriza por qué tan bien sigue una organización los principios del MDD y qué grado de automatización utiliza para soportar esos procesos.

Aunque [20] no trata un modelo de madurez, sí describe un modelo de calidad que considera niveles, junto con un conjunto de características que pueden ser evaluadas y medidas como maneras de determinar transiciones entre niveles.

El modelo descrito en [21] es una propuesta surgida dentro de un gran proyecto Europeo denominado MODELWARE. Su orientación es maximizar los beneficios de MDD para reducir tiempos de producción y su propósito es ayudar a las organizaciones a adoptar un enfoque MDD que les permita lograr automatización de procesos y capitalice el conocimiento del negocio en modelos reusables.

El artículo en [22] rescata un propósito fundamental de MDD, automatizar la generación de código desde los modelos a través de las transformaciones sucesivas de abstracciones modeladas en un nivel superior (dominio del problema) en modelos más concretos (dominio de la solución). Su propuesta es diseñar un proceso MDD que cumpla totalmente con un proceso de madurez de software bien definido, se presenta un enfoque MDD específico denominado OO-METHOD soportado en una herramienta comercial llamada OLIVANOVA. La valoración de cumplimiento se hace usando el método SCAMPI pero se restringe a un área de Proceso, la de Solución Técnica (TS).

El trabajo de [23] parte de la importancia que tiene el modelado de requerimientos en el desarrollo de software y menciona un enfoque de ingeniería de requerimientos, al que propone integrar como entrada en el contexto de un enfoque MDD, para producir, por transformación automática, un modelo inicial que luego será refinado para generar código a través de más procesos de transformación. Para incrementar la adopción de la propuesta se sugiere alinear los procesos definidos con un modelo de madurez, tal como el CMMI-DEV, el ejemplo de conformidad se hace con el área de proceso Desarrollo de Requerimientos (RD).

Ninguno de los trabajos relacionados ofrece el desarrollo de un modelo completo de madurez y capacidad con la parte de referencia de procesos y el modelo o método de evaluación, sólo los últimos [22], [23] hacen una aproximación restringida a una única área de procesos y considerando frameworks o herramientas específicas que limitan la aplicación del modelo.

V. CONCLUSIONES Y TRABAJO FUTURO

La aplicación del modelo fue hecha de forma parcial, a pesar que inicialmente hubo motivación y apoyo por parte de la dirección del área de informática, las presiones y urgencias de las entregas y de los compromisos dio una baja prioridad a las acciones del plan de mejora de procesos. De todas maneras, aunque pequeños, hubo avances en ciertos grupos de trabajo en metas específicas.

El modelo es aplicable a la gran mayoría de pequeñas y muy pequeñas empresas de desarrollo de software, incluyendo las que cuentan con experiencia en metodologías ágiles orientadas al código, pues MDD comprende también modelos textuales, tal como los lenguajes específicos de dominio DSL.

Sin embargo, es imprescindible obtener compromiso total de los niveles directivos para asignar recursos y tiempo del personal en la ejecución de los planes de mejora y apoyo para la institucionalización de los procesos. Así mismo para proveer recursos en herramientas y capacitación apropiada al desarrollo dirigido por modelos.

REFERENCIAS

[1] P. Naur and B. Randell, "Software Engineering: Report of a conference sponsored by the NATO Science Committee, Garmisch, Germany, 7-11 Oct. 1968, Brussels, Scientific Affairs Division, NATO," 1969.

[2] M. Jorgensen, "Failure factors of small software projects at a global outsourcing marketplace," *Journal of Systems and Software,* vol. 92, pp. 157-169, 2014.

[3] C. Y. Laporte, S. Alexandre and A. Renault, "Developing international standards for very small enterprises," *Computer,* vol. 41, no. 3, pp. 98-101, 2008.

[4] A. Arora and A. Gambardella, "The globalization of the software industry: perspectives and opportunities for developed and developing

countries," 2004.

[5] S. Cortezia and Y. Souza, "Aprendizagem na internacionalização de micro e pequenas empresas da indústria de software," *Anais do ENANPAD,* 2007.

[6] W. S. Humphrey, "Introduction to software process improvement," 1993.

[7] J. Bézivin, "On the unification power of models," *Software and Systems Modeling,* vol. 4, no. 2, pp. 171-188, 2005.

[8] S. Mellor, T. Clark and T. Futagami, "Model-driven development: guest editors' introduction.," *IEEE software,* vol. 20, no. 5, pp. 14-18, 2003.

[9] C. P. Team, "CMMI for Development Version 1.3 (CMMI-DEV, V. 1.3) CMU," 2010.

[10] I. 15504-5:2012, "Information Technology--Process Assessment--Part 5: An exemplar Process Assessment Model," Geneva, Switzerland, 2012.

[11] H. Oktaba, C. A. Esquivel, A. S. Ramos, A. M. Martínez, G. Q. Osorio, M. R. López, F. L. L. Hinojo, M. E. R. López, M. J. O. Mendoza, Y. F. Ordóñez and others, *Modelo de procesos para la industria de software moprosoft,* Versi{\'o}n, 2003.

[12] F. Pino, F. García and M. Piattini, "Revisión sistemática de mejora de procesos software en micro, pequeñas y medianas empresas," *Revista Española de Innovación, Calidad e Ingeniería del Software,* vol. 2, no. 1, pp. 6-23, 2006.

[13] M. Montoni, A. Rocha and K. Weber, "MPS. BR: a successful program for software process improvement in Brazil," *Software Process: Improvement and Practice,* vol. 14, no. 5, pp. 289-300, 2009.

[14] G. Valdés, H. Astudillo, M. Visconti and others, "The Tutelkan SPI Framework for small settings: A methodology transfer vehicle," *Systems, Software and Services Process Improvement,* pp. 142-152, 2010.

[15] J. Hurtado, F. Pino, J. Vidal, C. Pardo, L. Fernandez, M. Piattini and H. Oktaba, "Agile SPI: Software Process Agile Improvement, A Colombia approach to software process improvement in small software organizations," *Software process improvement for small and medium enterprises: techniques and case studies (M. Piattini, Oktaba, H.). Idea Group Inc,* vol. 1, no. 1, pp. 177-192, 2008.

[16] V. S. Hernández, "La Industria Del Software. Estudio A Nivel Global Y América Latina," *Observatorio de la Economía Latinoamericana,* no. 116, 2009.

[17] J. Warmer and A. Kleppe, The object constraint language: getting your models ready for MDA, Addison-Wesley Professional, 2003.

[18] S. Huang and S. Tilley, "Towards a documentation maturity model," in *Proceedings of the 21st annual international conference on Documentation*, 2003.

[19] Unisys, "Moving up the model-driven development maturity curve," 2008.

[20] C. Lange and M. Chaudron, "Managing model quality in UML-based software development," in *Software Technology and Engineering Practice, 2005. 13th IEEE International Workshop on*, 2005.

[21] E. Rios, T. Bozheva, A. Bediaga and others, "MDD maturity model: a roadmap for introducing model-driven development," in *Model Driven Architecture--Foundations and Applications*, 2006.

[22] A. M. Lins de Vasconcelos, G. Giachetti, B. Marín and O. Pastor, "Towards CMMI-compliant MDD Software Processes," in *ICSEA 2011, The Sixth International Conference on Software Engineering Advances*, 2011.

[23] A. M. L. de Vasconcelos, G. Giachetti, B. Marín and O. Pastor, "Towards a CMMI-Compliant Goal-Oriented Software Process through Model-Driven Development," in *The Practice of Enterprise Modeling*, Springer, 2011, pp. 253-267.

[24] N. Bouleau, "La modélisation et les sciences de l'ingénieur," *NOUVEL, P. Enquéte sur,* 2002.

[25] G. Cernosek and E. Naiburg, "The value of modeling," *IBM developerWorks,* 2004.

[26] P. Chen, "The entity-relationship model toward a unified view of data," *ACM Transactions on Database Systems (TODS),* vol. 1, no. 1, pp. 9-36, 1976.

[27] M. B. Albizuri-Romero, "A retrospective view of CASE tools adoption," *ACM SIGSOFT Software Engineering Notes,* vol. 25, no. 2, pp. 46-50, 2000.

[28] J. Sztipanovits and G. Karsai, "Model-integrated computing," *Computer,* vol. 30, no. 4, pp. 110-111, 1997.

[29] G. Booch, J. Rumbaugh and I. Jacobson, Unified Modeling Language--User{'}s Guide, Addison-Wesley Reading, MA, 1999.

[30] S. Mellor, MDA distilled: principles of model-driven architecture, Addison-Wesley Professional, 2004.

[31] E. Seidewitz, "What models mean," *Software, IEEE,* vol. 20, no. 5, pp. 26-32, 2003.

[32] C. Gibson and R. Nolan, "Managing the four stages of EDP growth," *Hypothesis,* p. 399, 1973.

[33] W. E. Deming, "Out of the crisis. Cambridge, MA: Massachusetts Institute of Technology," *Center for Advanced Engineering Study,* p. 6, 1986.

[34] W. A. Shewhart, "Economic control of quality of manufactured product," 1931.

[35] P. B. Crosby, Quality is free: The art of making quality certain, vol. 94, McGraw-Hill New York, 1979.

[36] G. O'Regan, Introduction to software process improvement, Springer, 2011.

[37] J. Iivari, "A paradigmatic analysis of information systems as a design science," *Scandinavian Journal of Information Systems,* vol. 19, no. 2, p. 39, 2007.

[38] J. Pfeffer and R. Sutton, The knowing-doing gap: How smart companies turn knowledge into action, Harvard Business Press, 1999.

[39] J. Bach, "The Immaturity of the CMM," *American Programmer,* vol. 7, pp. 13-13, 1994.

[40] J. D. Herbsleb and D. R. Goldenson, "A systematic survey of CMM experience and results," in *Software Engineering, 1996., Proceedings of the 18th International Conference on*, 1996.

[41] I. T. 29110-1, "Software Engineering - Lifecycle Profiles for Very Small Entities (VSEs) - Part 1: Overview," Geneva, Switzerland, 2011.

[42] S. Garcia, "Thoughts on applying CMMI in small settings," *Software Engineering Institute,* 2005.

[43] M. Fowler, "The new methodology," *Zugriff,* vol. 4, p. 2007, 2005.

[44] M. Fowler and J. Highsmith, "The agile manifesto," *Software Development,* vol. 9, no. 8, pp. 28-35, 2001.

[45] H. Glazer, J. Dalton, D. Anderson, M. D. Konrad and S. Shrum, "Cmmi or agile: Why not embrace both," 2008.

[46] S. Ambler and M. Lines, Disciplined Agile Delivery: A Practitioner's Guide to Agile Software Delivery in the Enterprise, IBM Press, 2012.

[47] I. 12207:2008, "Systems and software engineering -- Software life cycle processes," Geneva, Switzerland, 2008.

[48] P. COMPETISOFT, *COMPETISOFT-Mejora de Procesos para Fomentar la Competitividad de la Pequeña y Mediana Industria del Software de Iberoamérica. Versión 1.0,* CYTED, 2008.

[49] T. J. Maurer and H. R. Pierce, "A comparison of Likert scale and traditional measures of self-efficacy.," *Journal of applied psychology: American Psychological Association,* p. 324, 1998.

[50] L. J. Cronbach, "Coefficient alpha and the internal structure of tests," *Psychometrika,* vol. 16, no. 3, pp. 297-334, 1951.

978-1-4799-6717-9/14/$31.00 ©2014 IEEE